Appl Intell (2013) 38:226–238
DOI 10.1007/s10489-012-0367-7

A resource enhanced HTN planning approach for emergency decision-making

Zhe Wang · Hong-Wei Wang · Chao Qi · Jian Wang

Published online: 3 August 2012

Abstract Hierarchical resource reasoning is one of the key issues to successfully apply Hierarchy Task Network (HTN) planning into emergency decision-making. This paper proposes a Resource Enhanced HTN (REHTN) planning approach for emergency decision-making with the objective to enhance the expressive power and improve the processing speed of hierarchical resource reasoning. In the approach, resource timelines are defined to describe various resource variables and constraints. Top-down resource reasoning is used for decomposing the resource constraints of upper-level tasks into those of lower-level tasks. Meanwhile, resource and temporal constraints of tasks in different branches are processed by causal links. After the tasks are decomposed into primitive tasks, resource profiles of consumable resources and reusable resources are checked by separate resource allocation processes. Furthermore, a constraint propagation accelerator is designed to speed up hierarchal resource reasoning. The effectiveness and practicability of REHTN are confirmed with some experiments from emergency logistics distribution problems.

Z. Wang · H.-W. Wang (✉) · C. Qi · J. Wang
Institute of Systems Engineering, Huazhong University of Science and Technology, Wuhan, China
e-mail: hwwang@mail.hust.edu.cn

Z. Wang
e-mail: philowang1980@gmail.com

C. Qi
e-mail: qichao@mail.hust.edu.cn

J. Wang
e-mail: wj-hust-2006@163.com

H.-W. Wang
Key Laboratory of Education Ministry of Image Processing and Intelligent Control, Huazhong University of Science and Technology, Wuhan, China

Keywords HTN planning · Resource reasoning · Hierarchical resource · Emergency decision-making · Emergency logistics distribution

1 Introduction

Resource reasoning is significant in many emergency decision-making issues, particularly the issues containing the coupling relationship between planning and scheduling. Resource, as a temporal concept, impacts emergency decision-making processes deeply [1]. In fact, some emergency action plans become invalid due to resource and temporal constraints. Moreover, large-scale emergency response, which contains complicated hierarchical task selection processes, increases the difficulty to handle emergency resources. For example, emergency logistics distribution, which is a fundamental emergency decision-making issue, requires more sophisticated response effects than general business logistics in resource analysis and reasoning [2]. The analysis of temporary transportation modes in emergencies, from tactical (e.g., road transport and waterway transport) to operational (e.g., car and truck) levels, means hierarchical resource constraints are included in the domain knowledge of emergency response. Unpredictable resource demand leads to the diversity of emergency resources. At the same time, large-scale emergency logistics results in a considerable computation amount.

Hierarchy Task Network (HTN) planning is widely used in emergency decision-making for its powerful ability in expressing and reasoning domain-specific knowledge [3–6]. However, HTN planning still does not have sufficient power to handle the domain knowledge with hierarchical resource constraints, since it lacks an explicit mechanism for hierarchical resource reasoning [7, 8]. Our research interest lies

in developing the reasoning ability of HTN planner to handle hierarchical resources in emergency decision-making. In order to satisfy the diversity of emergency resources, the expressive power of hierarchical resource reasoning needs to be enhanced for HTN planning [9–11]. Furthermore, responders who underlie psychological pressure and limited response time in emergency decision-making are difficult to provide the accurate and perfect domain knowledge [12–14]. It is far more complicated to consider the cases in which emergency resource and temporal constraints are inconsistent, or even conflicted [15]. Therefore, hierarchical resource reasoning should include modifying incomplete resource states into complete resource states. This leads to a long response time of planning, considering the large amount of computation of emergency response. For this reason, the processing speed of resource reasoning must be increased to satisfy response time requirements in practical emergency decision-making. In summary, the expressive power and processing speed of hierarchical resource reasoning are two challenges of successfully applying HTN planning into emergency decision-making.

On one hand, a mechanism is needed to deal with hierarchical resource reasoning for task networks. Currently, in AI planning there are two families of approaches dealing with resource reasoning: the state-oriented approach and the time-oriented approach [16, 17]. The state-oriented approach uses explicit global resource states, and the approach can directly verify whether the plan is feasible. For example, the HTN planner O-Plan2 [18] exploits a resource utilization manager (RUM) using optimistic and pessimistic resource profiles in the planning to achieve state-oriented resource reasoning. However, O-Plan2 does not deal with hierarchical resource constraints in task networks which exist in emergency decision-making. By contrast, the time-oriented approach adopts resource and time functions to implicitly complete resource reasoning. Various temporal reasoning techniques, such as timelines, temporal assertions, and chronicles [19, 20], are devised to describe time variables in the time-oriented approach. Resource requirements are described as functions of time variables. Then, the time-oriented approach represents the planning procedure as a set of partial specified resource and time functions describing the revolution of local resource profile states. Thus, the time-oriented approach can deal with complicated local resource requirements such as relative temporal constraints and persistent conditions. The time-oriented approach, however, cannot directly verify whether the resource states are feasible. It is well known that HTN planning is a state-based forward planning. Thus, utilizing the techniques of the time-oriented approach might improve the expressive power of resource reasoning in HTN planning, but a temporal enhancement mechanism for HTN planning must be devised to verify the feasibility of the planning. An example of temporal enhancements in HTN planning is SIADEX [21]. However, the temporal enhancements of SIADEX are devoted to primitive tasks, without concerning the resource constraints of tasks in different branches of task networks.

On the other hand, the search space of HTN planning increases greatly if complex resource and temporal constraints are taken into account. In order to increase the processing speed of resource reasoning, constraint satisfaction techniques can be introduced into HTN planning to reduce the domains of variables to satisfy the requirements of response time for practical emergency decision-making [22, 23]. Constraint satisfaction techniques are powerful in solving combinatorial optimization problems [24]. The basic idea of these techniques is to model problems as multi-valued state constraints and design a general model to solve them [25–27]. One of the difficulties of combining AI planning and constraint satisfaction techniques is that AI planning has an uncertain length of tasks but constraint satisfaction techniques require a certain number of variables. Thus, specific constraint models need to be designed to integrate AI planning and constraint satisfaction techniques. The temporal planner IxTeT [28] adopts a minimal critical set (MCS), which is based on constraint programming, to deal with resource conflicts. Its architecture, however, makes its ability to express domain-specific knowledge weaker than most of HTN planners. SIADEX [29] uses a constraint propagation engine-PC2 to reduce the domain of temporal variables. However, the engine can only be used for temporal constraints and primitive tasks, and cannot handle with hierarchical resource and temporal constraints in task networks.

In this paper, we propose a Resource Enhanced HTN (REHTN) planning approach to manage hierarchical resources of task networks in emergency domain with the aim to enhance the expressive power and to increase the processing speed of hierarchical resource reasoning. Firstly, the resource and temporal constraints in task networks, either the constraints within a complex task or the constraints of tasks in different branches, are defined as functions on resource timelines. Secondly, for primitive tasks, REHTN automatically converts the resource functions on resource timelines into global resource states by uniform resource allocation and inspection. Finally, a constraint propagation accelerator on resource timelines is designed to speed up hierarchical resource reasoning in task networks. REHTN combines the advantages of both the state-oriented approach and the time-oriented approach. Hence, emergency responders can focus on various local resource and time requirements. Meanwhile, REHTN can automatically verify whether the global resource states satisfy the resource and time requirements of HTN planning.

The remainder of this paper is organized as follows. The primary architecture of REHTN is introduced in Sect. 2.

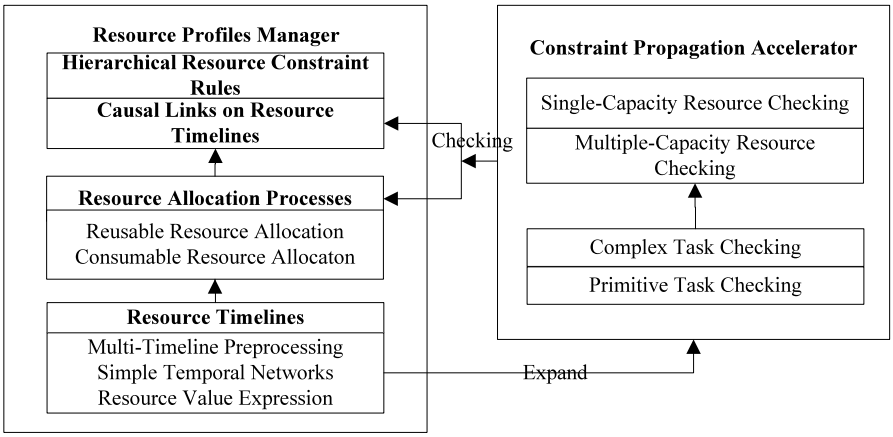

Fig. 1 The primary architecture of REHTN

Section 3 presents the mechanism of hierarchical resource reasoning in task networks. Furthermore, the constraint propagation accelerator is designed to speed up hierarchical resource reasoning in Sect. 4. In Sect. 5, an experimental study of emergency logistics distribution problem is used to validate the effectiveness of REHTN. This paper is concluded in Sect. 6.

2 Primary architecture of REHTN

Based on awareness and recognition of domain knowledge, HTN planning decomposes complex tasks recursively into lower-level tasks to realize the specific goal formula, until the set of tasks selected and organized by the planning is entirely composed of primitive tasks. Resource reasoning for emergency decision-making needs dynamically allocating resources and time in the processes of task analysis and reasoning. Hierarchical resource reasoning is the unique feature of HTN planning, because it differs from resource reasoning of non-HTN planning in considering top-down reasoning. Hierarchical resource reasoning can obtain as accurate hierarchical resource information as possible. Additionally, hierarchical resource information can help HTN planning evaluate the tasks in task networks to improve the quality of the plan.

Resource is a widely used concept with fuzzy boundary in literature. Hierarchical resource reasoning in the proposed REHTN deals with capacity resources, whose quantity can be normalized as numeric values. Numeric computation occupies most of computation amount in decision-making processes, and thus the analysis and reasoning of capacity resources can effectively support emergency decision-making. Figure 1 shows the primary architecture of REHTN. REHTN extends HTN planning by adding two major components: resource profiles manager aiming at establishing the centralized management of hierarchical resource reasoning, and constraint propagation accelerator aiming at improving the processing speed of hierarchical resource reasoning. On one hand, the resource profiles manager focuses on handling the interactions between resources and tasks, and thus capacity resources can be divided into consumable resources and reusable resources for deliberate consideration according to the resource requirements of tasks. On the other hand, the constraint propagation accelerator aims at increasing the processing speed, and thus capacity resources can be classified into single-capacity resources and multiple-capacity resources for the accelerating algorithm. Figure 2 depicts the planning procedure of REHTN.

To strengthen the centralized management of hierarchical resource reasoning in emergency environment, the resource profiles manager needs to identify various hierarchical resource constraints, and then validate the consistency of these constraints uniformly. Resource timelines, which are the expansion of Simple Temporal Networks (STN) [25], are designed to describe hierarchical resources in the resource profiles manager [30, 31]. By defining the independent resource timeline for each kind of resources, the resource profiles manager can deal with multi-resource constraints. In the planning procedure, the resource profiles manager represents the resource and temporal constraints as functions on resource timelines, and then propagates these constraints to primitive tasks. For this purpose, top-down resource reasoning is used to decompose the resource functions of upper-level tasks into those of lower-level tasks following hierarchical resource constraint rules (Line 25, Fig. 2). Meanwhile, causal links on resource timelines are established to represent the resource and temporal constraints of tasks in different branches (Lines 20, 31, Fig. 2). Moreover, resource allocation processes of reusable resources and consumable resources allocate the variables of the resource functions on resource timelines to primitive tasks, and then update the global resource states (Line 12, Fig. 2). In essence, the processes realize the transformation from the resource functions to the global resource states.

Fig. 2 The planning procedure of REHTN

```
1:  Procedure REHTN (sr, T, P, R)
2:   set P = null , the plan;
3:   set R , the resource list that the domain is related;
4:   set sr , the initial current global resources states;
5:   set T_0 ← { t ∈ T | no other task in T precedes t };
6:   loop
7:   if T = ∅ then return (P, R) ;
8:     choose any t ∈ T_0 ;
9:     generate the control variables of t on resource timelines;
10:    if t is a primitive task, then
11:    choose one ground instance of t ;
12:    execute the resource allocation processes of t ;
13:      if sr satisfies t 's preconditions then
14:      insert the control variables of t to R
15:      execute the constraint propagation accelerator on resource timelines ( R );
16:        if all variables' domain in R is not null then
17:          sr ← sr + add(t) − delete(t) ;
18:          insert t to P ;
19:          delete t from T ;
20:          identify the causal links on resource timelines of t ;
21:          else delete the control variables of t from R ;
22:      else there is no more primitive tasks satisfying sr return FAIL;
23:    else t is complex task, then
24:    choose one of t 's decomposition methods of whose preconditions are fit for
sr ;
25:    add the hierarchical resource constraint rules;
26:    set R', the resource list that including the control variables from t and its
subtask {t_1, t_2, ....., t_n} ;
27:    execute the constraint propagation accelerator on resource timelines ( R' );
28:      if all variables' domain in R is not null then
29:      insert {t_1, t_2, ....., t_n} to P ;
30:      delete t from T ;
31:      identify the causal links on resource timelines of t ;
32:    else there is no more decomposition methods for t return FAIL;
33:    repeat;
34:    end REHTN.
```

In order to reduce redundant computation, the constraint propagation accelerator based on resource timelines devises an incremental method to deal with complex tasks and primitive tasks. On one hand, for the decomposition process of a complex task, the constraint propagation accelerator detects the consistency of the complex task and its sub-tasks (Lines 26, 27, Fig. 2). On the other hand, for the choosing process of primitive tasks, the constraint propagation accelerator detects the consistency of all primitive tasks (Lines 14, 15, Fig. 2). The constraint propagation accelerator adopts the similar concepts used in PC-2 algorithm [32] due to its comprehensive performance of universality and algorithm complexity. In addition, resource constraint propagation algorithms, such as edge-finding and energetic reasoning [26], can also be used in constraint propagation accelerator. Unfortunately, the strict assumed conditions of these algorithms about rigid resource variation and time window make these algorithms unsuitable for emergency decision-making.

According to the domain knowledge, REHTN can generate task sequences with resource allocation information. Therefore, with the help of REHTN, emergency decision support system (EDSS) can effectively solve actual resource-constrained emergency decision-making issues. The two major components of REHTN are discussed in more details in the next two sections.

3 Resource profiles manager

To realize the centralized management of hierarchical resource reasoning in REHTN, the resource information including type, location, and quantity in task networks is requested to be verified in the resource profiles manager. Traditional resource reasoning only devotes to primitive tasks [18, 29], but hierarchical resource reasoning needs to deal with task networks. For the purpose of facilitating centralized management, resource timelines are designed to ex-

press the resource and temporal constraints in task networks. The resource and temporal constraints, either the constraints within a complex task or the constraints of tasks in different branches, can be defined as functions on resource timelines. In order to validate whether the plan is feasible, the functions on resource timelines are converted into the global resource states for primitive tasks.

3.1 Resource description

The resource profiles manager utilizes resource timelines to describe various resource and temporal constraints in task networks. Two types of resources, consumable resources and reusable resources, are described as separate functions on resource timelines. The definitions of resources are given as follows:

Definition 1 Let $R = (\mathrm{Re}\,sourceID, \mathrm{Re}\,sourceTYPE, RT)$ be a resource list, where Re *sourceID* marks all the resources involved in the domain knowledge and Re *sourceTYPE* is the set of resource types. Let $RT = (X, D, C, Q, M, N)$ be the resource timeline for each kind of resource, where (X, D, C) is the structure of STN [25]; (Q, M, N) is the resource property structure based on time variables; X is the set of discrete time variables; D is the set of domains of time variables; C is the set of temporal constraints of time variables; Q is the set of resource instantaneous variations corresponding time points in X; M is the set of domains of resource instantaneous variations; and N is the set of resource constraints of resource instantaneous variations.

The resource profiles manager records all reference time points. The global current time CT is defined in a plan. Each task t in the plan owns two time points t_start and t_end corresponding to the start time and the end time of the task, respectively. Further, the time constraint $t\_duration = t\_end - t\_start$, in which $t\_duration$ is the duration time of task t, should be satisfied. All of the time points share the same domain $[D\min, D\max]$. In addition, the early start time of task t is the minimum value of the domain of t_start; meanwhile, the deadline of task t is the maximum value of the domain of t_end. To facilitate the transformation from multi-valued resource variables to resource states, REHTN only considers the resource variations that happen at the start or end instants of tasks. Furthermore, the resource constraints of temporal intervals can be transformed into temporal instants by Allen's algebra [29, 33].

The resource profiles manager deals with consumable resources and reusable resources. A consumable resource is a resource which is used up or produced by a task [16]. Each consumable resource re has a resource timeline RT_{re}, an initial value INI_{re}, a current time CT_{re}, and a current value CQ_{re}. The capacity of re is in $[Q\min_{re}, Q\max_{re}]$, and the shared domain of time variables on RT_{re} is $[D\min_{re}, D\max_{re}]$. The resource usage $t\_\cos t$, one of the resource properties, is designed to record the usage at time t_start. In contrast, a reusable resource is a resource which is "borrowed" by a task, and released afterward [16]. Each reusable resource rr also has a resource timeline RT_{rr}, an initial value INI_{rr}, and a current time CT_{rr}. The capacity of rr is in $[Q\min_{rr}, Q\max_{rr}]$, and the shared domain of time variables on RT_{rr} is $[D\min_{rr}, D\max_{rr}]$. The allocation $t\_allocation$ and deallocation $t\_deallocation$ of resource rr are designed to record the resource amounts of occupation at t_start and release at t_end, respectively.

3.2 Top-down resource reasoning of task decomposition

The resource profiles manager realizes top-down resource reasoning by propagating the resource and temporal constraints of upper-level tasks into those of lower-level tasks in decomposition process. For this purpose, the resource profiles manager automatically adds certain conditions for the preconditions of sub-tasks following hierarchical resource constraint rules to realize top-down reasoning of consumable resources and reusable resources. There are three different types of task decomposition structures in HTN planning: sequential structure, unordered structure and optional structure [34–36]. A planning tree is an And/Or tree that represents the set of all possible decomposition branches of a task network [7]. Figure 3 is the planning tree, which illustrates the three structures, and Fig. 4 is the corresponding diagram of task decomposition. This section analyzes the hierarchical resource constraint rules of these three types of structures.

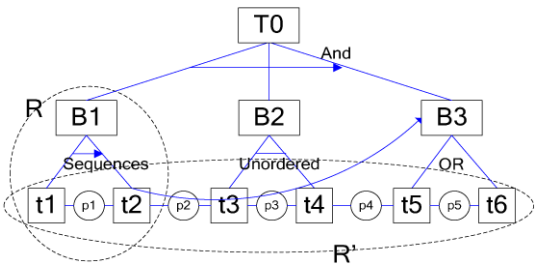

Fig. 3 A planning tree illustration including three types of task decomposition structures

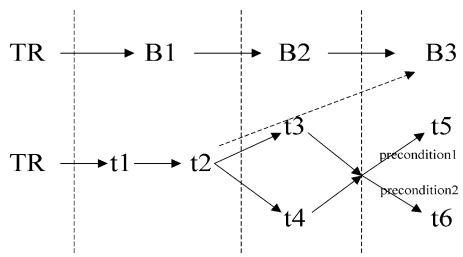

Fig. 4 A graphical representation of task decomposition

(1) For sequential structure

In the sequential structure, the sequence of sub-tasks comes from task decomposition. An example of sequential structure is shown as follows:

(task $B1$ Parameters:(...) ResourceVariables:(...)

Preconditions:(...) Sub-tasks:($t1$ $t2$))

In task decomposition, sub-tasks $t1$ and $t2$ need to inherit all the resource and temporal constraints of $B1$. The temporal constraint $0 \leq t2\_start - t1\_end \leq +\infty$ should be satisfied. Moreover, for consumable resource re, if the resource constraint of $B1$ is $Q1 \leq B1\_\cos t \leq Q2 (Q1, Q2 \in [Q\min_{re}, Q\max_{re}], Q1 \leq Q2)$, and $t1$ satisfies the resource constraint $Q1 \leq t1\_\cos t \leq Q2$, then the resource constraint $Q1 - t1\_\cos t \leq t2\_\cos t \leq Q2 - t1\_\cos t$ should be added for $t2$. In contrast, since the reusable resource allocated to $t1$ has been released at the start of $t2$, it is unnecessary to add more resource constraints.

(2) For unordered structure

In the unordered structure, the sequence of sub-tasks is not ordered by task decomposition. An example of unordered structure is shown as follows:

(task $B2$ Parameters:(...) ResourceVariables:(...)

Preconditions:(...) Sub-tasks:(:unordered $t3$ $t4$))

In task decomposition, sub-tasks $t3$ and $t4$ need to inherit all the resource and temporal constraints of $B2$. Moreover, for consumable resource re, if the resource constraint of $B2$ is $Q1 \leq B2\_\cos t \leq Q2$ $(Q1, Q2 \in [Q\min_{re}, Q\max_{re}], Q1 \leq Q2)$, then the resource constraint added for $t3$ and $t4$ is represented as $Q1 \leq t3\_\cos t + t4\_\cos t \leq Q2$. Meanwhile, for reusable resource rr, if the resource constraint of $B2$ is $Q1 \leq B2\_allocation \leq Q2$ $(Q1, Q2 \in [Q\min_{rr}, Q\max_{rr}], Q1 \leq Q2)$, and the occupation and release of resources of the other sub-tasks in unordered structure before the time point $t3\_start$ are $t3\_this\_allocation$ and $t3\_this\_deallocation$. Then, the added resource constraint for $t3$ is $Q1 \leq t3\_allocation + t3\_this\_allocation - t3\_this\_deallocation \leq Q2$. Similarly, the added constraint for $t4$ is $Q1 \leq t4\_allocation + t4\_this\_allocation - t4\_this\_deallocation \leq Q2$.

(3) For optional structure

In the optional structure, sub-tasks are optional according to the precondition of complex task. An example of optional structure is shown as follows:

(task $B3$ Parameters:(...) ResourceVariables:(...)

Case1 Preconditions1:(...) Sub-tasks:($t5$)

Case2 Precondition2:(...) Sub-tasks:($t6$))

In task decomposition, sub-task $t5$ and $t6$ inherit all the resource and temporal constraints of $B3$.

3.3 Resource reasoning of tasks in different branches based on causal links

In task networks of emergency decision-making, there exist many resource constraints of tasks in different branches, such as the constraint between $t2$ and $B3$ in Fig. 3. Causal links on resource timelines are introduced in the resource profiles manager to deal with hierarchical resource and temporal constraints of tasks in different branches. Causal links firstly invent for partially ordered planning, represent the relation that a producer task is intended to give the propositions for a consumer task [19]. The superiorities of utilizing causal links to represent the resource and temporal constraints are twofold. Firstly, the completeness of resource reasoning is ensured. Secondly, redundant tasks are avoided effectively because the resources which belong to the producer and consumer task wouldn't be moved back and forth caused by other tasks.

Definition 2 Let $CL = (a_i, a_j, RT, P(RT), C(X_{a_i}, X_{a_j}), N(Q_{a_i}, Q_{a_j}))$ be a causal link on resource timeline, where a_i is the producer task; a_j is the consumer task; RT is the resource timeline associated with a_i and a_j; $P(RT)$ is the proposition set which a_i produces for a_j, $C(X_{a_i}, X_{a_j})$ is the set of temporal constraints between a_i and a_j, where X_{a_i} and X_{a_j} are the set of time points on RT for a_i and a_j, respectively; $N(Q_{a_i}, Q_{a_j})$ is the set of resource constraints between a_i and a_j, where Q_{a_i} and Q_{a_j} are the set of resource variables on RT for a_i and a_j, respectively.

Causal links on resource timelines need to deal with the resource and temporal constraints of complex tasks. To represent the literals of causal links, it is significant to introduce the effects of complex tasks, while complex tasks have no effects in existing HTN planners [9, 10, 37]. In order to meet the requirement, the high-level effects [16] in REHTN, which cannot update any states including current resource states, are designed to be only used in protected conditions. Figure 5 illustrates the causal links on resource timelines between $t2$ and $B3$, with RT belonging to the consumable resource re. The positive literals of protected conditions mean any task that changes the states in protected conditions cannot be executed. On the contrary, the negative literals of protected conditions cancel the protection. Thus, REHTN can identify causal links on resource timelines by protected conditions.

For each complex task, REHTN assumes that the positive literals of protected conditions are executed before its sub-tasks and the negative literals of protected conditions are executed after its sub-tasks. In Fig. 5, for consumable resource re, the temporal constraint $0 \leq B3\_start - t2\_end \leq +\infty$ is added to $C(X_{t2}, X_{B3})$, and the resource constraint $Q1 \leq B3\_\cos t + t2\_\cos t \leq Q2 (Q1, Q2 \in$

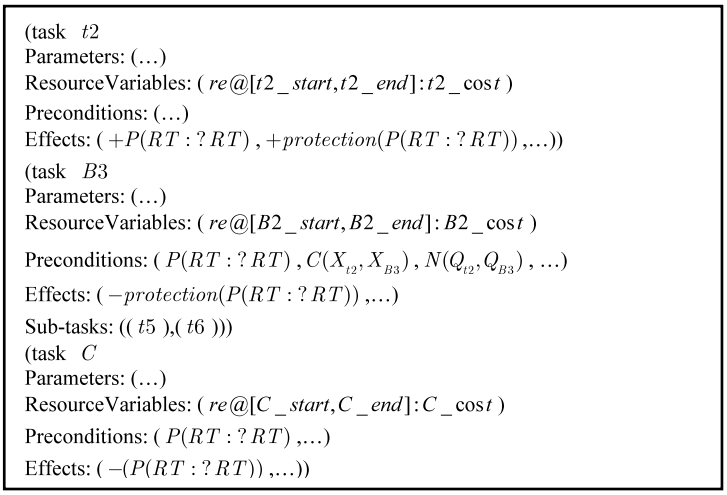

Fig. 5 An illustration of causal link on resource timeline: $(t2, B3, RT, P(RT), C(X_{t2}, X_{B3}), N(Q_{t2}, Q_{B3}))$ threatened by C

$[Q\min_{re}, Q\max_{re}]$, $Q1 \le Q2$) is added to $N(Q_{t2}, Q_{B3})$; for reusable resource rr, $t2$ is executed before $B3$, then it just need to add the temporal constraint $0 \le B3\_start - t2\_end \le +\infty$. When $B3$ is decomposed to $t5$ and $t6$, $C(X_{t2}, X_{B3})$ and $N(Q_{t2}, Q_{B3})$ are propagated to $t5$ and $t6$. The processes continuously cycle in the decomposition of $B3$ until $B3$ is entirely decomposed into primitive tasks.

3.4 Resource profiles verification

The resource profiles manager utilizes resource allocation processes to validate resource profiles. The resource allocation processes allocate the variables of the resource functions on resource timelines to primitive tasks, generate the general resource constraints for primitive tasks, and then update the global resource states. For consumable resources, the resource profiles at the start points of primitive tasks in the plan are checked. For reusable resources, the resource profiles between the start and end points of primitive tasks in the plan are checked. The resource allocation processes for consumable resources and reusable resources are analyzed in particular as follows.

(1) For consumable resources

Consumable resource re cannot exceed the range bounded by the minimum capacity and the maximum capacity. For primitive task $A1$, the start time is $A1\_start$, the end time is $A1\_end$, and the usage amount of consumable resource re is $A1\_\cos t$. If $A1\_\cos t$ is positive, the task consumes the resource. On the contrary, if $A1\_\cos t$ is negative, the task produces the resource. The following verification processes are executed by REHTN.

Step 1: For the preconditions of $A1$, generate and inspect the following general resource constraints:

$$Q\max_{re} \ge CQ_{re} - A1\_\cos t \ge Q\min_{re},$$
$$D\max_{re} \ge A1\_end \ge A1\_start \ge D\min_{re},$$
$$A1\_start \ge CT_{re}.$$

Step 2: For the preconditions of $A1$, inspect the special resource and temporal constraints of $A1$. Using the variables on resource timelines, the resource and temporal constraints of $A1$ are easily identified. In addition, the resource and temporal functions follow the standard in PDDL2.2. For more details, please refer to the literature [38].

Step 3: For the preconditions of $A1$, inspect the resource and temporal constraints inherited from the upper-level tasks of $A1$.

Step 4: For the effects of $A1$, generate the effects to change the values of CQ_{re}, CT_{re}, and CT to $CQ_{re} - A1\_\cos t$, $A1\_end$, and $\max(CT, A1\_end)$, respectively.

(2) For reusable resources

Reusable resource rr differs from consumable resource re in considering recovery. For each task $A2$, the start time $A2\_start$ corresponding to the occupation of resource $A2\_allocation$ and the end time $A2\_end$ corresponding to the release of resource $A2\_deallocation$ are given on RT_{rr}. The following verification processes are executed by REHTN.

Step 1: For the preconditions of $A2$, generate and inspect the following general resource constraints:

$$Q\max_{rr} \ge INI_{rr} - \left(\sum_{t\ge 0}^{t\le CT_{rr}} allocation - \sum_{t\ge 0}^{t\le CT_{rr}} deallocation\right) - A2\_allocation \ge Q\min_{rr},$$
$$D\max_{rr} \ge A2\_end \ge A2\_start \ge D\min_{rr},$$

Fig. 6 Outline of constraint propagation accelerator on resource timelines

1: Constraint propagation accelerator on Resource Timelines (π, t, R)
2: Let π be a partial resource enhanced HTN plan, and t be the current task of planning procedure
3: Let $R = (\mathrm{Re}\,sourceID, \mathrm{Re}\,sourceTYPE, RT)$ be a resource list.
4: Let $C(X_t)$ be the temporal constraint set of t, $N(Q_t)$ be the resource constraint set of t.
5: Let $X(C(X_t))$ be the set of discrete time variables associated to $C(X_t)$, $Q(N(Q_t))$ be the set of resource instantaneous variations associated to $N(Q_t)$.
6: Let $P \leftarrow \{(i,k,j), 1 \le i < j \le \mid X(C(X_t)) \mid, 1 \le k \le \mid X \mid, k \ne i \ne j\}$
7: If $R \ne \phi$, then nondeterministically select and remove a tuple $r = (\mathrm{Re}\,sourceID, \mathrm{Re}\,sourceTYPE, RT)$ from R.
8: If $P \ne \phi$, then
9: Select and remove a path (i,k,j) from P
10: If $C_{ij} = \mathrm{Re}\,vise(i,k,j)$ has changed after $\mathrm{Re}\,vise(*)$ then $P \leftarrow P \cup RELATED\_PATHS(i,k,j)$
11: Until C_{ij} is stabilization
12: If r is a single-capacity resource, then exit
13: Else if r is a multiple-capacity resource, then
14: Let $S \leftarrow \{(u,w,v), 1 \le u < v \le \mid Q(N(Q_t)) \mid, 1 \le w \le \mid Q \mid, u \ne v \ne w\}$
15: Select and remove a path (u,w,v) from S
16: If $N_{uv} = \mathrm{Re}\,vise(u,w,v)$ has changed after $\mathrm{Re}\,vise(*)$ then $P \leftarrow P \cup RELATED\_PATHS(u,w,v)$
17: Until N_{uv} is stabilization
18: **End accelerator**
$\mathrm{Re}\,vise(i,k,j): C_{ij} = C_{ij} \cap (C_{ik} \circ C_{kj})$ $\mathrm{Re}\,vise(u,w,v): N_{uv} = N_{uv} \cap (N_{uw} \circ N_{wv})$

$A2\_start \ge CT_{rr}$,

$A2\_allocation = A2\_deallocation$.

Step 2: For the preconditions of $A2$, inspect the special resource and temporal constraints of $A2$.

Step 3: For the preconditions of $A2$, inspect the resource and temporal constraints inherited from the upper-level tasks of $A2$.

Step 4: For the effects of $A2$, generate the effects to change the value of CT_{rr} and CT to $A2\_end$ and $\max(CT, A2\_end)$, respectively.

4 Constraint propagation accelerator

The constraint propagation accelerator on resource timelines in REHTN speeds up hierarchical resource constraint propagation in task networks by pruning the search space. The idea of the constraint propagation accelerator on resource timelines is adapted from similar concepts used in PC-2 [32] to eliminate the inconsistency of resource and temporal constraints. Figure 6 shows the rough outline of the constraint propagation accelerator on resource timelines.

PC-2 is a path consistency algorithm with constraint propagation operation. Constraint propagation operation propagates the constraints of variable set to its adjacent variables, and eliminates redundant values and tuples. The operation recurs constantly, until a fix point appears. The path consistency of three variables (x_i, x_k, x_j) means, for any value (v_i, v_j) satisfying the constraint C_{ij}, there is a sequence of values (v_k) such that (v_i, v_k) satisfies the constraint C_{ik} and (v_j, v_k) satisfies the constraint C_{jk}. It is noted that, if (x_i, x_k, x_j) are path consistent, then $RELATED\_PATHS(i,k,j)$ represents all paths that include (i,j) or (j,i).

The constraint propagation accelerator on resource timelines designs incremental methods for complex tasks and primitive tasks. On one hand, for the decomposition process of a complex task, the variable set of the complex task and its sub-tasks, such as R in Fig. 3, is detected by constraint propagation operation (Lines 26, 27, Fig. 2). On the other hand, for choosing the process of primitive tasks, the variable set of all primitive tasks, such as R' in Fig. 3, is detected by constraint propagation operation. (Lines 14, 15, Fig. 2). The variables of hierarchical resources are easily identified by resource timelines. Let t be the current task of the planning, then $X(C(X_t))$ is the set of all discrete time variables associated with $C(X_t)$, and $Q(N(Q_t))$ is the set of all resource variables associated with $N(Q_t)$. Adopting incremental methods, the constraint propagation accelerator

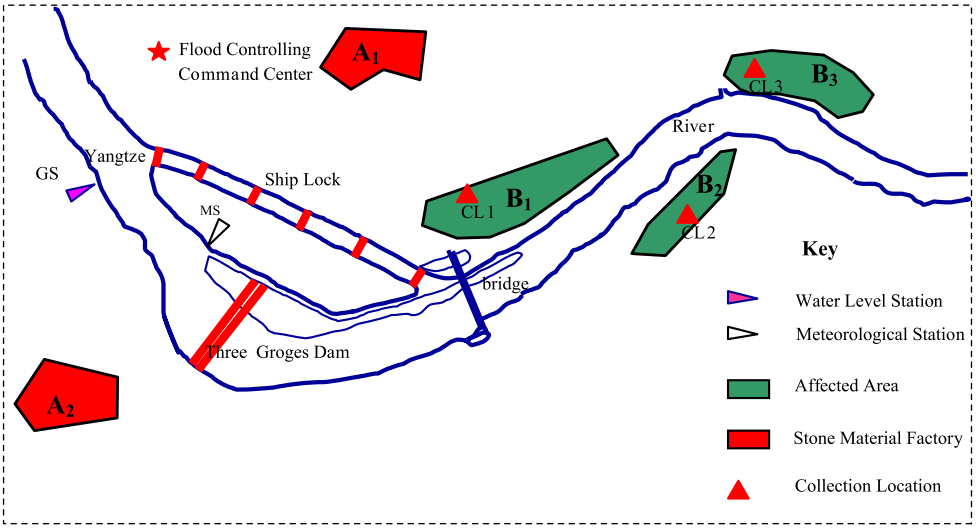

Fig. 7 A simplified map of the flood response

checks whether the variables associated with the current task are consistent with all variables on resource timelines.

For single-capacity resources, the constraint propagation accelerator checks the temporal consistency; for multiple-capacity resources, the accelerator also checks the resource consistency besides the temporal consistency. The idea of solving the resource and temporal consistency separately might result in an incomplete algorithm. However, the completeness of hierarchical temporal reasoning in REHTN is checked by the resource profiles manager. As a consequence, the aim of the constraint propagation accelerator is the acceleration of hierarchical resource constraint propagation in task networks without checking the completeness of resource and time variables. Moreover, in emergency decision-making, responders can analyze the resource and temporal requirements separately. Therefore, solving the resource and temporal consistency separately is feasible.

5 Experimental study for emergency logistics distribution

As one of the most severe natural disasters in China, flood disaster often takes place in the vast region around lakes and rivers. In response to flood disasters, large amount of emergency resources are allocated and transported in limited time for disaster relief, evacuation-and-transfer, search-and-rescue, resettlement, rehabilitation, and other issues [1]. Thus, emergency logistics distribution is one of most important decision-making issues responding to flood disasters. Figure 7 shows a hypothetical emergency scene of flood response for the Three Gorges Dam. Different from general business logistics, emergency logistics distribution is featured by the coupling relationship of hierarchical task selection and hierarchical resource reasoning. Simplifying the decision-making processes of stone traffic programs in the China Three Gorges Corporation, the emergency logistics distribution domain is devised in order to validate the effectiveness of REHTN.

5.1 Emergency logistics distribution domain

The emergency logistics distribution domain involves transporting emergency materials to satisfy series of materials demands at affected areas by different transportation modes. In order to guarantee the effectiveness of emergency response, some safety measures are taken, such as reserving emergency materials and assigning co-drivers. After simplifying the emergency logistics in the China Three Gorges Corporation, this domain can be described as follows: the regions $A_1, A_2, \ldots, A_n$ are the suppliers (e.g., stone material factories), and A_i stores emergency materials (e.g., stone, geotextile and steel pipe) $X(A_i)$ tons ($X(A_i) > 0, i \in \{1, 2, \ldots, n\}$). The regions $B_1, B_2, \ldots, B_m$ are the demanders (e.g., affected areas). Each region B_i ($i \in \{1, 2, \ldots, m\}$) triggers l rescue tasks on the time $t(B_i)_j$ ($j \in \{1, 2, \ldots, l\}$), and expends materials $Y(B_i)_j \times k(B_i)_j$ % tons ($k(B_i)_j \in [0, 100]$). Each region B_i needs to keep $Y(B_i)_j$ tons of materials before the time $t(B_i)_j$. After the time $t(B_i)_j$, the superfluous materials of B_i can be used to satisfy the requirement of other regions. The temporal constraints of $t(B_i)_j$ is represented as a Simple Temporal Problem [23]. In addition, the constraint $\sum X(A_i) \geq \sum_i \sum_j Y(B_i)_j \times k(B_i)_j$ should be satisfied and the total incurred cost should not be more than M RMB. The distance between the regions is denoted by $d(x, y)$ kilometers, such that $x, y \in \{A_1, A_2, \ldots, A_n, B_1, B_2, \ldots, B_m\}$.

Emergency transportation has two options:

(1) The *delivery* method. This method entrusts transport tasks to transport teams. $TEAM_1, TEAM_2, \ldots, TEAM_{nt}$ are transport teams. The maximum transport capacity

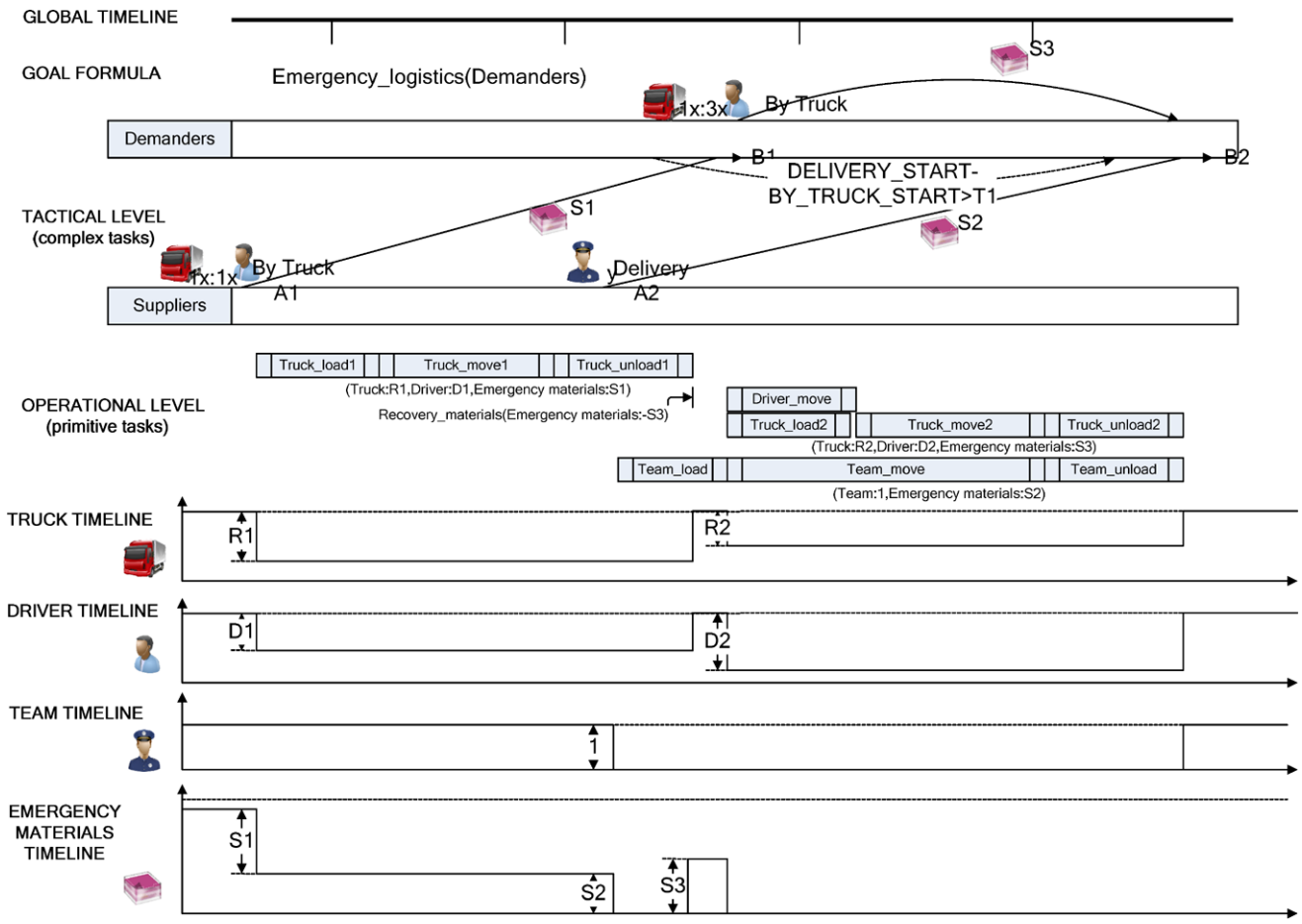

Fig. 8 The graphical illustration of an emergency logistics distribution example

of each team $TEAM_i$ $(i \in \{1, 2, \ldots, nt\})$ is $CAPACITY_i$ tons. The speed of $TEAM_i$ is $SPEED_i$ kilometer/hour. The initial position of $TEAM_i$ as the same as its final position is $LOC_i \in \{A_1, A_2, \ldots, A_n\}$. The billing formula of $TEAM_i$ is $SPEND + K\_TEAM_i \times THIS\_CAPACITY$ RMB/kilometer, such that $THIS\_CAPACITY$ is the cargo volume for each transport task.

(2) The *by truck* method. This method organizes trucks and drivers for transport tasks. To ensure safety, one truck requires more than one driver. For simplicity, we do not discuss the difference between drivers and co-drivers. If there are not enough drivers in the local region, it is necessary to deploy drivers from other regions in order to drive trucks. The deployment of drivers needs costs and waiting time. The amount of trucks in region A_i is $u(A_i)$ and the number of drivers is $v(A_i)$. The speed of the truck is $SPEED\_TRUCK$ kilometer/hour, and the maximum capacity of the truck is $CAPACITY\_TRUCK$ tons. The billing formula of trucks is $SPEND\_TRUCK + K\_TRUCK \times THIS\_CAPACITY$ RMB/kilometer, such that $THIS\_CAPACITY$ is the cargo volume for each transport task. The deployment of k drivers needs T_DRIVER hours and costs $W\_DRIVER \times k$ RMB. In a transport task to region B_i, each truck needs $L(B_i)$ drivers to ensure safety. Loading /unloading materials needs $h \in [h\min, h\max]$ hours, and costs $SPEND\_LOAD - K\_LOAD \times h$ RMB.

If no-load vehicles are not taken into account, the *delivery* method incurs higher unit price than the *by truck* method, but is safer and more efficient. However, in order to complete the transportation in time, no-load vehicles are necessary in some cases. More complicated, the domain considers the deployment of drivers and the recovery of emergency materials. Thus, it is difficult to determine which transportation means is cheaper for each task by mathematical model. In contrast, a feasible plan can be obtained from REHTN by describing the domain knowledge.

In this domain, teams, trucks, and drivers are reusable resources, whereas emergency materials and money are consumable resources. From another perspective, teams are single-capacity resources, whereas trucks and drivers are multiple-capacity resources. In addition, location and quantity of emergency materials should be considered simultaneously. Figure 8 shows the tasks and timelines in an emergency logistics distribution example. In the example, the

Table 1 Comparison between resource profiles manager and mixed symbolic/numeric computation in emergency logistics distribution domain

Problem number		ES1	ES2	ES3	ES4	ES5	ES6	ES7	ES8	ES9	ES10
Resource profiles manager	Cost (RMB)	4200	11625	16755	27705	36206	43405	52705	61005	68205	87930
	CPU (ms)	1069	1436	1872	3010	3978	4992	5990	7036	8284	9313
mixed symbolic/numeric computation	Cost (RMB)	9400	15990	34565	46255	53905	61255	72505	85310	92660	98740
	CPU (ms)	655	1123	1810	2168	2449	2699	2946	3400	3682	4402

Table 2 Comparison between REHTN and JSHOP2 in emergency logistics distribution planning domain

Problem number		ELD1	ELD2	ELD3	ELD4	ELD5	ELD6	ELD7	ELD8	ELD9	ELD10
REHTN	*Cost (RMB)*	4500	12225	17755	29005	37905	45605	55205	63905	71505	91730
	CPU (ms)	782	1134	1810	2168	2456	2899	4245	5038	6849	8035
JSHOP2	*Cost (RMB)*	4500	13700	211100	30300	39600	48100	56300	65450	74855	92655
	CPU (ms)	1060	1436	1872	3010	3978	4992	5990	7036	8284	9313

goal formula is recorded as *Emergency_logistics*, which means accomplishing all of the demanders' resource requisition. To achieve the goal formula, the resource and time variables of emergency logistics tasks can be conducted by resource timelines. For example, the early start time of emergency logistics is encoded as the minimum value of the start time point *Emergency_logistics_start*. Meanwhile, the deadline of emergency logistics is encoded as the maximum value of the end time point *Emergency_logistics_end*. In the tactical level, REHTN can help emergency responders choose transportation means. According to the unit price, the *by truck* method is chosen for the materials demands in the demander B_1. Furthermore, relative temporal constraints (e.g., the occupation time of trucks is T_1 hours earlier than that of teams) are conducted by causal links. According to top-down resource reasoning, the resource and temporal functions of the demander B_2 is gained. Analyzing the functions, the materials demands in the demander B_2 is achieved by transporting from the supplier A_2 and recovering from the demander B_1. Further, complex tasks of tactical level are decomposed into primitive tasks of operational level. In the operational level, resources are allocated to the primitive tasks with resource profiles verification, and thus, the specific recourse timelines are reprocessed. For example, the primitive task *Driver_move* is added to the plan because the resource allocation process detects the resource shortage of local drivers and chooses *Driver_move* to make up this shortage. The final resource timelines are expressed in Fig. 8.

5.2 Experimental results

The experimental study is carried out on a Pentium IV PC with Core 2 Duo T5750 and 2 GB RAM in Windows XP. We develop REHTN using Java language by expanding the grammar of SHOP2 as it is one of the most popular HTN planners [10].

Table 1 presents a comparison between resource profiles manager and mixed symbolic/numeric computation [10]. The experimental results indicate the efficiency of hierarchical resource reasoning. The resource profiles manager realizes hierarchical resource reasoning. By contrast, the mixed symbolic/numeric computation does not achieve hierarchical resource reasoning but can calculate resource levels at certain time points in the plan [10]. The two techniques are realized in REHTN. Because the mixed symbolic/numeric computation cannot deal with relative temporal constraints in task networks, we simplify the emergency logistics distribution domain to compare the resource profiles manager with the mixed symbolic/numeric computation. We randomly construct ten groups of simple emergency logistics distribution problems which do not involve relative temporal constraints in a flood disaster situation denoted by $\{ES1, ES2, \ldots, ES10\}$. The results indicate that the total incurred cost of the resource profiles manager is less than that of the mixed symbolic/numeric computation. However, the CPU time of the resource profiles manager is longer than that of the mixed symbolic/numeric computation because of hierarchical resource reasoning. The experimental results validate that hierarchical resource reasoning can improve the quality of the plan but with the cost of more CPU time. For this reason, it is important to design the constraint propagation accelerator of REHTN.

Table 2 list the experimental results comparing REHTN with JSHOP2. JSHOP2 uses the Multi-Timeline Preprocessing scheme (MTP) to explicitly encode the relative temporal constraints [10]. We randomly test ten groups of emergency logistics distribution problems in a flood disaster sit-

uation denoted by $\{ELD1, ELD2, \ldots, ELD10\}$. The total incurred cost of REHTN is less than that of JSHOP2, because REHTN identifies more resource requirements and selects cheaper transportation means in limited time. Meanwhile, the CPU time of REHTN is shorter than that of JSHOP2 because REHTN adopts the constraint propagation accelerator to speed up resource constraint reasoning. The results indicate that REHTN is effective for the problems with hierarchical resource requirements.

6 Conclusions and future work

Hierarchical resource reasoning, which is the unique feature of HTN planning, can help us effectively solve the domain knowledge with hierarchical resource constraints in emergency decision-making environment. This paper proposes REHTN for emergency decision-making with the aim to enhance the expressive power and to improve the processing speed of hierarchical resource reasoning. By designing the resource timelines, REHTN can express more realistic hierarchical resource requirements in task networks. By converting resource functions to global resource states, REHTN checks whether the plan is feasible. Moreover, the constraint propagation accelerator on resource timelines is designed to improve the efficiency of REHTN. The experimental study is conducted to validate the effectiveness of REHTN.

Our on-going works will carried out in two directions. Firstly, there are some cases that hierarchical resource requirements cannot be satisfied in emergency decision-making environment, which can be regarded as partial satisfaction planning problems. REHTN can be improved to deal with partial satisfaction planning problems in emergency decision-making. Secondly, there are some cases that resource requirements and resource states constantly change in emergency decision-making environment, and thus, the dynamic performance of REHTN can be improved.

Acknowledgements The authors are grateful to Mr. Jingjing Li for the domain knowledge modeling of emergency logistics distribution problem. We would also like to express our thanks to Dr. Yongchang Wei and Dr. Pan Tang for numerous discussions of this paper. This work was supported by the National Science Foundation Grant of China, projects No. 90924301, No. 91024032 and No. 71125001.

References

1. Altay N, Green W (2006) OR/MS research in disaster operations management. Eur J Oper Res 175(1):475–493
2. Sheu J (2007) An emergency logistics distribution approach for quick response to urgent relief demand in disasters. Transp Res, Part E, Logist Transp Rev 43(6):687–709
3. Asunción M, Castillo L, Fdez-Olivares J, Garcıa-Pérez O, González A, Palao F (2005) Knowledge and plan execution management in planning fire fighting operations. In: Planning, scheduling and constraint satisfaction: from theory to practice. IOS Press, Amsterdam, pp 149–158
4. Biundo S, Bercher P, Geier T, Muller F, Schattenberg B (2011) Advanced user assistance based on AI planning. Cogn Syst Res 12(3–4):219–236
5. Kuzu M, Cicekli N (2012) Dynamic planning approach to automated web service composition. Int J Appl Intell 36(1):1–28. doi:10.1007/s10489-010-0238-z
6. Nau D, Tsz-Chiu A, Okhtay I (2005) Applications of SHOP and SHOP2. IEEE Intell Syst 20(5):34–41
7. Biundo S, Schattenberg B, Ghallab M, Hertzberg J, Traverso P (2002) On the identification and use of hierarchical resources in planning and scheduling. In: Proceedings of the 6th international conference on artificial intelligence planning systems (AIPS-2002), Toulouse, France, pp 263–272
8. Smith DE, Frank J, Honsson AK (2000) Bridging the gap between planning and scheduling. Knowl Eng Rev 15(1):47–83
9. Asuncion M, Castillo L, Fdez-Olivares J, Garcia-Perez O, Gonzalez A, Palao F (2005) SIADEX: an interactive knowledge-based planner for decision support in forest fire fighting. AI Commun 18(4):257–268
10. Nau D, Tsz-Chiu A, Okhtay I (2003) SHOP2: an HTN planning system. J Artif Intell Res 20(12):379–404
11. Sierbra C (2004) A unified approach to planning support in hierarchical coalitions. PhD Thesis, www.aiai.ed.ac.uk/project/ix/project/siebra/resources/SiebraThesis.pdf, Artificial Intelligence, Applications Institute, The University of Edinburgh
12. Benton J, Do M, Kambhampati S (2009) Anytime heuristic search for partial satisfaction planning. Artif Intell 173(5–6):562–592
13. Potter S (2011) Critical reasoning: AI for emergency response. Int J Appl Intell. doi:10.1007/s10489-011-0331-y
14. Sapena O, Onaindia E (2008) Planning in highly dynamic environments: an anytime approach for planning under time constraints. Int J Appl Intell 29(1):90–109
15. Bonet B, Geffner H (2001) Planning and control in artificial intelligence: a unifying perspective. Int J Appl Intell 14(3):237–252
16. Ghallab M, Nau D, Traverso P (2004) Automated planning: theory and practice. Elsevier, Amsterdam
17. Bacchus F, Kabanza F (2000) Using temporal logic to express search control knowledge for planning. Artif Intell 116(1–2):123–191
18. Drabble B, Tate A (1994) The use of optimistic and pessimistic resource profiles to inform search in an activity based planner. In: Proceedings of the international conference on AI planning systems (AIPS-94), Chicago, USA, pp 243–248
19. Schattenberg B (2009) Hybrid planning and scheduling. PhD Thesis, http://vts.uni-ulm.de/docs/2009/6895/vts_6895_9580.pdf, Institute of Artificial Intelligence, Ulm University
20. Verfaillie G, Praclet C, Lemaitre M (2010) How to model planning and scheduling problems using constraint networks on timelines. Knowl Eng Rev 25(3):319–336
21. Castillo L, Fdez-Olivares J, Garc Ma-Perez O (2006) Temporal enhancements of an HTN planner. In: Lecture notes in computer science. Springer, Berlin, pp 429–438
22. Garrido A, Barber F (2001) Integrating planning and scheduling. Appl Artif Intell 15(5):471–491
23. Sapena O, Onaindia E, Garrido A, Arangu M (2008) A distributed CSP approach for collaborative planning systems. Eng Appl Artif Intell 21(5):698–709
24. Mouhoub M, Sukpan A (2012) Conditional and composite temporal CSPs. Int J Appl Intell 36(1):90–107. doi:10.1007/s10489-010-0246-z
25. Dechter R, Meiri I, Pearl J (1991) Temporal constraint networks. Artif Intell 49(1–3):61–95
26. Laborie P (2003) Algorithms for propagating resource constraints in AI planning and scheduling: existing approaches and new results. Artif Intell 143(2):151–188

27. Nareyek A, Freuder E, Fourer R, Giunchiglia E, Goldman R, Kautz H, Rintanen J, Tate A (2005) Constraints and AI planning. IEEE Intell Syst 20(2):62–72
28. Lemai S, Ingrand F (2003) Interleaving temporal planning and execution $I_XT_ET_EXEC$. In: Proceedings of international conference on automated planning and scheduling (ICAPS-03) workshop on plan execution, Trento, Italy, pp 1–10
29. Castillo L, Fdez-Olivares J, Garc Ma-Perez O (2006) Efficiently handling temporal knowledge in an HTN planner. In: Proceedings of international conference on automated planning and scheduling (ICAPS-06), Cumbria, UK, pp 63–72
30. Penna G, Magazzeni D, Mercorio F (2012) A universal planning system for hybrid domains. Int J Appl Intell 36(4):932–959. doi:10.1007/s10489-011-0306-z
31. Yaman F, Nau D (2002) Timeline: an HTN planner that can reason about time. In: Proceedings of the 6th international conference on artificial intelligence planning systems (AIPS-02) workshop on planning for temporal domains, Toulouse, France, pp 75–81
32. Alan K (1977) Consistency in networks of relations. Artif Intell 8(1):99–118
33. Allen J (1983) Maintaining knowledge about temporal intervals. Commun ACM 26(1):832–843
34. Erol K, Hendler J, Nau D (1994) HTN planning: complexity and expressivity. In: Proceedings of the national conference on artificial intelligence (NCAI-94), Seattle, USA, pp 1123–1128
35. Erol K, Hendler J, Nau D (1994) UMCP: a sound and complete procedure for hierarchical task-network planning. In: Proceedings of the international conference on AI planning system (AIPS-94), Chicago, USA, pp 249–254
36. Erol K, Hendler J, Nau D (1996) Complexity results for HTN planning. Ann Math Artif Intell 18(1):69–93
37. Tate A, Drabble B, Kirby R (1994) O-Plan2: an open architecture for command, planning and control. In: Intelligence scheduling. Morgan Kaufmann, San Francisco, pp 213–239
38. Edelkamp S, Hoffmann J (2004) PDDL2.2: the language for the classical part. In: 4th international planning competition (IPC-04), Whistler, Canada

Zhe Wang is a Ph.D. candidate at Institute of Systems Engineering, Huazhong University of Science and Technology, Wuhan, China. He obtained his M.Sc. and B.Sc. degrees in control science and engineering at the same university, in 2007 and 2003. His research interests include HTN planning, classical planning and constraint satisfaction techniques.

Hong-Wei Wang is a professor and the dean of the Department of Control Science and Engineering, Huazhong University of Science and Technology, Wuhan, China. He also was a senior visiting scholar at the University of Oxford, UK, in 2002. He obtained his Ph.D. degree in systems engineering from Huazhong University of Science and Technology, in 1993. His research interests covers artificial intelligence, public security and emergency management, and modeling and simulation of complex systems.

Chao Qi is an associate professor in the Department of Control Science and Engineering, Huazhong University of Science and Technology, Wuhan, China. She also was a senior visiting scholar at the Singapore-Massachusetts Institute of Technology Alliance, Singapore and USA, in 2006–2008. She obtained her Ph.D. degree in systems engineering and management from Nanyang Technological University, Singapore, in 2006. Her research interest includes AI planning, production planning and scheduling, and maintenance scheduling.

Jian Wang is an associate professor in the Department of Control Science and Engineering, Huazhong University of Science and Technology, Wuhan, China. He obtained his Ph.D. degree in systems engineering at the same university, in 2009. His research interest includes AI planning, emergency management, and modeling and simulation of complex systems.