Technological and Economic Development of Economy
ISSN: 2029-4913 / eISSN: 2029-4921
2019 Volume 25 Issue 3: 472–495
https://doi.org/10.3846/tede.2019.9384

HIERARCHICAL TASK NETWORK APPROACH FOR TIME AND BUDGET CONSTRAINED CONSTRUCTION PROJECT PLANNING

Dian LIU[1], Hong-wei WANG[2*], Heng LI[3], Johnny WANG[4], Mohamed KHALLAF[5]

[1]*School of Automation, Huazhong University of Science and Technology, Wuhan, China*
[2]*School of Management, School of Automation, Huazhong University of Science and Technology, Wuhan, China*
[3]*Department of Building and Real Estate, Hong Kong Polytechnic University, Hong Kong, China*
[4,5]*Faculty of Design, Architecture and Building, University of Technology Sydney, Sydney, Australia*

Received 20 May 2018; accepted 26 January 2019

Abstract. Completing a construction project on time and within budget is of great importance in the construction industry. To achieve this goal, a construction plan satisfying the time and cost constraints is crucial. While a rich amount of literature on the time-cost trade-off scheduling and time/cost optimization scheduling has been presented, developing a construction plan for the time and cost-constrained construction project has not been fully explored. This study presented a hierarchical task network (HTN) based construction planning model to fill this gap. First of all, a knowledge formalism catering to the HTN planning was provided to accommodate the construction planning knowledge. Then, the planning process was explained in detail, including temporal reasoning used to sequence the construction activities, and backtracking evasion mechanism used to avoid the trouble of backtracking due to inappropriate selection of execution modes for construction activities. Finally, two sets of comparisons based on a fictional construction project were performed, the results of which demonstrate that the time and budget constraints have an impact on the section of execution modes for construction methods, and the proposed planning model can develop construction plan that satisfies the specified deadline and budget limitations effectively regardless of the existing of backtracking.

Keywords: construction planning, project deadline, budget limitation, hierarchical task network planning.

JEL Classification: C63, D83, L74.

Introduction

Ensuring the construction project completed within a particular time frame, staying within budget and achieving all specific objectives (e.g. quality, safety performance) is considered the cornerstone of a successful project planning (Olawale & Sun, 2010). The construction

*Corresponding author. E-mail: *hwwang@mail.hust.edu.cn*

project management plan provides an important means to ensure all necessary works are completed within the required time so as to achieve the required project objectives and outcomes (Urizar & Halim, 2015). The process of developing a construction plan involves selecting the construction methods, defining the construction activities, identifying relationships among the different activities, and estimating the cost and duration of each activity (Hendrickson, Zozaya-Gorostiza, Rehak, Baracco-Miller, & Lim, 1987; Zozaya-Gorostiza, Hendrickson, & Rehak, 1989). The selection of construction methods would affect the production rate and unit cost to complete a construction activity, which provides information of the duration and cost required for the construction activity. As construction projects are often subject to certain constraints, e.g. time and cost, these constraints must be considered and analysed in the construction planning process. With the time and cost constraints together with temporal constraints between the construction activities, a careful selection of installation methods is crucial as each of the activities will affect each other in some way.

Traditional construction planning is time-consuming and error-prone. To alleviate these problems, the adoption of learning techniques to support complicated construction planning task has become an area of research interest. Numerous innovative tools and techniques, for example, case-based reasoning, knowledge-based approaches, model-based approaches, expert systems etc. have been adopted to automatically generate construction plans. A great deal of research efforts have also been made to optimize the construction schedules by reasonably arranging time and resources based on the given activity network (Faghihi, Nejat, Reinschmidt, & Kang, 2015; Vanhoucke, 2018; Zhou, Love, Wang, Teo, & Irani, 2013). While much progress has been made in automating the development of construction plans, many existing approaches performed limited ability of constraint reasoning to verify whether the construction plan satisfies the time and cost constraints. Many efforts for optimizing construction schedules assumed that the activity network has been determined, but didn't pay attention to the impacts of various construction methods on the structure of activity network (Abuwarda & Hegazy, 2016).In order to simultaneously consider the impact of construction methods on the activity network and project duration and cost, it is necessary to determine the construction methods in the process of generating the activity network while ensuring that the current and final construction plan meets the requirements of project duration and budget.

Hierarchical task network (HTN) planning is an automated planning technique in the field of artificial intelligence, which is able to utilize domain knowledge to decompose the abstract tasks that cannot be executed directly into smaller executable tasks in a top-down way, finally generating an action plan. Its powerful ability of reasoning makes it possible to handle time and cost constraints in the process of generating activity network (Liu, H. Wang, Qi, Zhao, & J. Wang, 2016; Nau, 2007; Qi, D. Wang, Muñoz-Avila, Zhao, & H. Wang, 2017). On the other hand, the extant functional decomposition of the construction process by trade specialists provides possibilities of using HTN to accomplish the construction planning task automatically (Kartam, Levitt, & Wilkins, 1991). Nevertheless, classical HTN planning technique is limited to choose an execution mode for an activity at a time, which will result in a large amount of backtracking when there are limitations on the quantitative parameters related to the selection of execution modes. This is because HTN planning performs a depth-first search, the planning process has to backtrack many search steps to the inappropriate option and makes another choice when quantitative constraints are not satisfied in deeper search

step due to an inappropriate selection of execution mode that happens in shallower search step. To adapt to the construction planning problem with a deadline and budget limitation, a new technique to deal with this deficiency is needed.

In this study, an HTN based construction planning model is proposed, which incorporates a backtracking evasion mechanism that postpones the selection of execution modes until all activities have been generated. Additionally, temporal reasoning technique is also integrated into the planning model to figure out the sequence relationships between different activities for the purpose of calculation project duration. In the subsequent sections, related work regarding construction planning, as well as the basic concepts of HTN planning, is first reviewed in Section 1. A formalism for describing the concerned construction planning problem is given in Section 2. The development of HTN-based construction planning model is presented in Section 3, and a case study is carried out to demonstrate the effectiveness of the approach in Section 4. Finally, we summarize the limitation of this study and outline prospective future work.

1. Literature review

In the literature, research regarding the development of construction plan can be divided into two categories: construction planning and construction scheduling. Construction planning cares about how to get an activity network from scratch that can complete the project, while construction scheduling emphasizes how to define the start time of activities in the network with or without resource constraints in the setting of single or multiple projects (An, Woo, Cho, & Lee, 2017; Kannimuthu, Ekambaram, Raphael, & Kuppuswamy, 2018). Since the investigated problem in this paper is concerned with how to generate an activity network that meets the time and cost requirement, related works regarding construction planning are first reviewed, including case-based reasoning and knowledge-based approach. Then, an introduction to HTN planning is given bravely.

1.1. Current case-based reasoning and knowledge-based approach to construction planning

Case-based reasoning (CBR) has been widely applied in construction management since the CBR and construction management problem solving shared a similar mind-sets (Hu, Xia, Skitmore, & Chen, 2016). When it comes to construction planning, CBR helps generate a schedule for a new project by applying the existing schedules of the similar past projects. Dzeng and Tommelein (1997, 2004) applied the CasePlan system to automatically generate new project schedule for the power plant boiler construction project from past project information. Tah, Carr, and Howes (1998, 1999) proposed a conceptual framework within which previous planning experiences can be captured and re-used to support the scheduling of new projects. K. J. Lee, Kim, J. K. Lee, and Kim (1998) developed a case- and constraint-based project-planning expert system for an apartment construction project. Zhang, Ding, and Love (2017) presented a modified CBR model with weighted k-nearest neighbors mechanism to help construction planners search for optimal similar plans with construction safety data, which are used to make new plans for deep foundation construction.

Despite these efforts, many of these approaches only limited the application to a specific type of construction projects, and are not applicable to other construction types. In this regard, Ryu, Lee, and Park (2007) presented a CBR-based general construction planning tool (CONPLA-CBR) for various types of construction projects, which consists of generic attributes to be customized to the given project. Xu and Muñoz-Avila (2008) presented a case-based planning prototype, CaBMA, that extracts experiences from previous project planning episodes and generalizes them into cases using HTN representation, then reuses these cases to complete partial project plans and generate new plans from scratch. Although CaBMA eliminates the need for revising process and thus improves the efficiency of the system to some extent, CaBMA requires the help of project manager to refine the tasks that have no similar cases to match them, which does not conform to the intention of this article to automatically generate construction plans.

On the other hand, researchers have also paid much attention to knowledge-based systems that can assist construction manager with planning and scheduling construction projects (Benjamin, Babcock, Yunus, & Kincaid, 1990). Hendrickson et al. (1987) established a knowledge-intensive expert system, CONSTRUCTION PLANEX, to accomplish the whole construction planning process relying on distinct knowledge sources of tables and rules specific to technology choices, duration estimation, or other consideration. Shaked and Warszawski (1995) developed an expert system, HISCHED, to generate construction plan using an object-oriented representation of building and production rules, routines, and functions for manipulating object attributes. Fischer and Aalami (1996) presented a model-based system for automated construction planning, which formalizes the construction methods using the computer-interpretable template to capture planning knowledge regarding construction activities, action sequence, constituting objects and resource requirements. Based on those method templates, high-level activities can be refined into appropriate lower-level activities.

Although these approaches have greatly improved the situation where manual development of construction plans is time-consuming and can lead to errors, the planning system that adopts those models cannot choose construction method according to site condition or other factors (e.g. duration, cost) because they do not specify the linkages between construction method and its corresponding construction activity. In contrast, relying on its powerful expressiveness for planning knowledge, HTN planning can record the reason behind the selection of construction method for a construction activity. In this respect, Kartam et al. developed a general-purpose planner SIPE (Kartam & Levitt, 1990) and its advanced version SIPE2 (Kartam et al., 1991) to support the automatic and interactive generation of hierarchical, nonlinear construction plans. However, both planners have a limitation in handling the reasoning about durations and resources associated with activities, which is critical to deal with time and cost constraints.

As to the construction planning problem in this study, it involves choosing appropriate construction method for each construction activity to meet the deadline and budget limits. To this end, the planning model should be able to figure out the sequence relationships between activities and search for feasible construction method combination that satisfies the budget constraint. Since traditional HTN planning technique lacks the required abilities, an advanced HTN-based construction planning model is proposed.

1.2. Hierarchical task network (HTN) approach

A typical hierarchical task network (HTN) planning problem (Ghallab, Nau, & Traverso, 2004) can be expressed by a tuple P = <s_0, T_0, D>, where s_0 is the initial planning state; T_0 is the initial task network consisting of several goal tasks to be completed; D = <O, M> is the planning domain knowledge, which consists of operator set O and method set M. An operator is an action template that formalizes a class of executable actions in the domain, and a method formally describes the prescription used to decompose a specific class of composite tasks. The problem representation involves the concepts of planning state, task network, operators and methods, the details of which are described as follows.

A planning state is the conjunction of a set of literals, each of which denotes a fact in the real world of concern. Thus, a planning state is often represented as $s = \{l_i\}$ indicating the facts hold in the state of the world. Literal l_i conforms to first-order logic and is often encoded as (p t_1 t_2 … t_n) where predicate p implies the meaning of a fact and term t_i is the related constants used to explain the fact. For example, to express the fact "The truck TRUCK_A is at Location_B", the literal (truck-at TRUCK_A LOCATION_B) will be added to the state.

A task network is an acyclic digraph that represents a set of tasks and precedence relations among those tasks. Tasks can be classified into primitive tasks that can be completed by actions, and abstract tasks that cannot be completed directly by actions. Both types of tasks can be expressed in the form of (t r_1 r_2 … r_n), in which t is a task symbol, and r_1, r_2, …, r_n are terms. If all the terms are constants, the task is ground, otherwise, the task is parametric. The precedence relations are binary and have the form of (R t_1 t_2) indicating the order between task t_1 and task t_2 is specified by R, such as before, after, and so on.

An operator is a 4-tuple of the form o = <:operator *head pre eff*> in which: operator is a keyword used to identify an operator, the *head* is a parametric task that represents a class of primitive tasks that can be completed by applying this operator; *pre* represents the precondition for the execution of this operator, which is the logical connection of a set of parametric literals and holds when the planning state can assign values to its parameters and make it logically true. *eff* = <*del, add*> is the effect after the execution of an instantiated operator, *del* and *add* respectively represents the deleted and added facts from/to the planning state.

A method is a tuple of the form m = <:method *head* pre_1 $subtasks_1$ pre_2 $subtask_2$ …> where: method is a keyword to identify an HTN method; *head* denotes the abstract tasks that can be decomposed by this method; pre_i represents the precondition of the decomposition, having the same semantic with that of an operator; $subtasks_i$ represents the smaller tasks that are decomposed from the head task, as well as the precedence relationships among them. Each pre_i is mutually exclusive, and thus the whole method expresses a set of decomposition rules with "IF… THEN…ELSE" structure or SWITCH branching structure.

For a typical HTN planning process (Figure 1), it usually starts with the initial planning state, the initial task network, and the planning domain knowledge, and then generates an action plan. A ground task, t, is first selected from the current task network (the initial task network at the beginning), and then search for an operator or an HTN method whose head is unified with t such that the instantiated head is identical to the ground task (Step 1). If an HTN method matches with t, then apply the method to decompose t and replace t in the

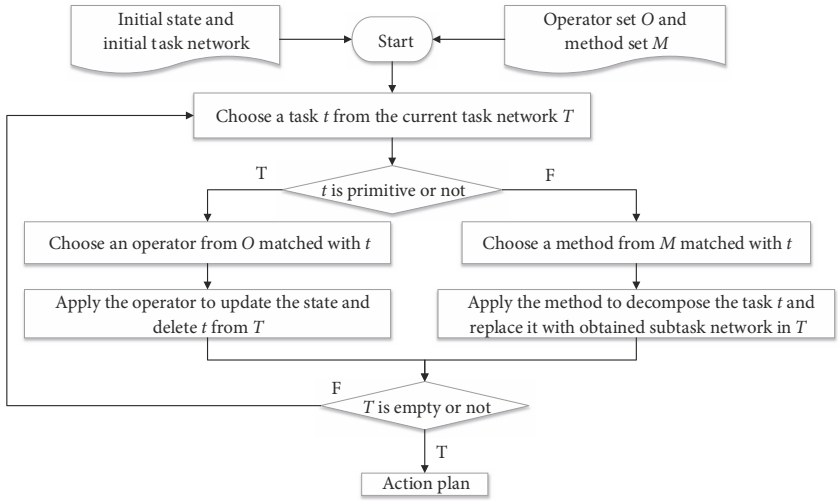

Figure 1. HTN planning process

task network with the method's subtasks (Step 2). If an operator matches with *t*, then apply the operator to change the planning state and delete *t* from the task network (Step 3). Then, one has to check whether the updated task network is empty. If it is 'yes', which means all the non-executable abstract tasks have been decomposed into executable primitive tasks and the action plan has been found, then the planning process ends. Otherwise, it repeats Steps (1)–(3) above.

2. A formalism for construction planning problem

To solve the construction planning problem using HTN planning, it is necessary to represent the construction planning problem in the above-mentioned formalism. However, the traditional HTN formalism doesn't contain elements for tackling deadline and other temporal features. Therefore, the above formalism is extended to fit the construction planning problem with a deadline and budget limits.

In this study, a construction planning problem is represented as a tuple $<CS_0, CG, CP_{Domain}>$ where CS_0 is the initial construction state that describes the *project-related information* in the form of literals, such as design model, site environment, geological conditions, among others; *CG = (cgTask, D, B)* is the construction goal which means completing the overall task *cgTask* within the deadline *D* and budget *B*; CP_{Domain} represents the *domain-dependent construction planning knowledge* regarding how to decompose the overall task and how to select the construction method. The term 'domain-dependent' means that those knowledge can be reused in different projects of the same construction domain (e.g. building, highway, railway, etc.). The following sub-sections introduce the representation of project-related information and domain-dependent construction planning knowledge in details.

2.1. Project-related information

Every construction project is unique and varies in design, size, capacity, location and site conditions, etc. The materials or components used would determine the types of construction activities required for the project. Geological conditions of the site (e.g. soil properties and the permeability coefficient) determines the choice of substructure construction methods be adopted. Each project generates different knowledge and information that would be useful for the construction planning.

Specifically, the product model of a construction project contains all the components need to be constructed, which is often represented as an object hierarchy (Fischer & Aalami, 1996; Hendrickson & Au, 1989; Hendrickson et al., 1987; Jägbeck, 1994), as shown in Figure 2. Nevertheless, the hierarchical model doesn't present the topological relationships among construction objects of different levels, which is critical to sequence the construction activities corresponding to those construction objects. To accommodate construction objects and topological relationships between them, a hierarchical product model consisting of an object set *OBJS* and a relation set *TR* is used here. Object set *OBJs* contains two types of construction objects: composite construction objects and primitive construction objects. Composite construction objects are those composed of other construction objects, such as Block and Floor in Figure 2, while primitive construction objects are those cannot be decomposed anymore, such as Design Element in Figure 2. Two types of construction objects can be encoded in the same form:

$$obj = \left(Type,\ obj\_id,\ name,\ parameters\right), \forall obj \in OBJs,$$

where *Type* indicates the type of the construction object, composite or primitive; *obj_id* is the unique identifier of the construction object; *name* provide the physical meaning of the construction object, such as Floor, Column, etc.; *parameters* are used to describe required information of the construction object, such the floor number of Floor, the volume of the Column.

Relationships in relation set *TR* can be divided into two categories: the subordinate relationship between construction objects of adjacent levels and the topological relationship among construction objects of the same level. Subordinate relationship is binary and has the form (*is_component_of obj_lower obj_supper*), indicating that the lower level construction object *obj_lower* is a component of the upper level construction object *obj_upper*.

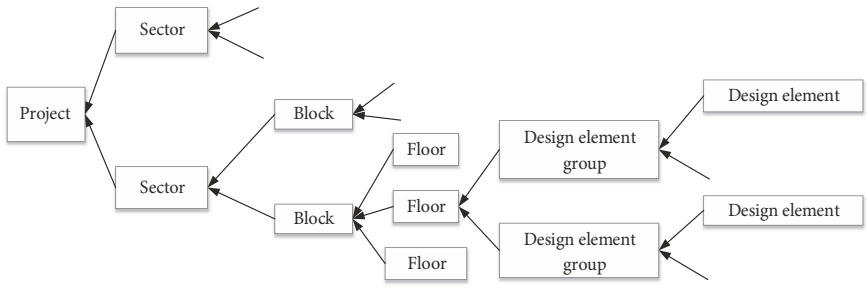

Figure 2. Object hierarchy of product model (Hendrickson et al., 1987)

In contrast, topological relationship may be binary or multivariate and can be expressed by the form $\left(R_{name}\ \ R_{member_1}\ R_{member_2}\ldots R_{member_n}\right)$, in which R_{name} represents the relationship type, $R_{member_i}\ (i=1,\ldots,n)$ denotes the interrelated construction objects of the relationship, and n is the number of interrelated construction objects, satisfying $n \geq 2$. As Echeverry, Ibbs, and Kim (1991) pointed out, there are different types of topological relationships among product elements that affect the sequencing of their corresponding activities. Examples of each are provided in Table 1.

Table 1. Selected examples of topological relationships

Relationship	Example	Expression
Supported by	(Slab of Second Floor) *SUPPORTED BY* (Columns of First Floor)	*(supported_by obj1 obj2)*
Covered by	(Foundation wall) *COVERED BY* (Bituminous Asphalt Waterproofing)	*(covered_by obj1 obj2)*
Embedded in	(Electrical Conduit) *EMBEDDED IN* (Stud Wall)	*(embedded_in obj1 obj2)*
Connect	(Stairs of First Floor) *CONNECT* (First Floor and Second Floor)	*(connect obj1 obj2 obj3 …)*
Enclosed by	(Masonry wall) *ENCLOSED BY* (Column and Slabs)	*(enclosed_by obj1 obj2 obj3 …)*

Apart from above information, some other information can also be similarly expressed in the initial construction state, including the site geological conditions, the engineering quantities of different component sets, and the production rate and unit cost of construction units. For example, the underground water level of a site can be encoded as (underground_water_level *?site ?depth*), and the fact can be used to select the construction method for groundwater control.

2.2. Domain-dependent knowledge

This kind of knowledge refers to those summaries that can be reused in a unique type of construction projects, such as construction codes, regulations, standards and manual. Corresponding to construction planning, that knowledge can be classified into two categories: i) knowledge for decomposing the abstract construction tasks, and ii) knowledge for defining construction activities.

First, to decompose the overall construction task along the hierarchical product model into executable construction tasks, knowledge describing how to perform the decomposition is required. Because there are several levels in the hierarchical product model, a generic expression is needed to represent this kind of decomposition while avoiding instantiated expressions for each subordinate relationship between construction objects of adjacent levels. Thus, the following tuple is used to express the knowledge for decomposing abstract tasks:

$$m_{decompse} = \langle task,\ pre,\ subtasks,\ constraints \rangle,$$

where *task* is encoded as (construct *?c_obj*), denoting an abstract construction task for the composite construction object *?c_obj*, *pre* describes the subordinate relationship between

?c_obj and the construction objects of the lower level; *subtasks* denotes the construction tasks for the subordinate construction objects of *?c_obj*; and *constraints* are used to express the precedence relationships between subtasks which are derived from the topological relationships between their corresponding construction objects.

Second, construction activities should be defined for the completion of primitive construction objects. Since more than one construction method is applicable to a construction object, the corresponding action can be performed in one of several alternative execution modes, each of which needs different time and cost. Meanwhile, sequence relationships between construction activities have a direct impact on the project completion time, which can be derived from the interactions between preconditions and effects of the execution of construction activities. For these reasons, the knowledge for define construction activities should contain elements for describing alternative execution modes and interaction between construction activities. A tuple of the following form is utilized to define construction activities:

$$CA = \langle head,\ pre,\ eff,\ ExM,\ BgDl \rangle,$$

where *head* = <*name, obj*> indicates this type of construction activity identified by *name* is able to complete the primitive construction object *obj*; *pre* represents the precondition of the activity's execution, such as the requirement of completing the supporting construction object for a supported construction object; *eff* indicates the changes to the construction state, such as the demolition of scaffolds and the completion of columns; *ExM* = <*amount, type*> is used to calculate duration and cost of the construction activity for the primitive object, in which *amount* is a variable representing quantitative amount of the primitive object and *type* is a variable representing the type of the construction activity, such as earthwork backfilling, cast-in-situ, etc. *BgDl* = <*budget, deadline*> expresses the budget and deadline limits used to examine whether the defined construction activity meet the time and cost limitations.

The above formalism is generic to the building construction domain. When it comes to the different building construction project, a construction planning problem can be encoded according to the generic formalism and is solved by the following construction planning process. A construction plan, as a result, is obtained to instruct the completion of the project.

3. Model development

In this section, a planning model is developed on the basis of HTN approach to solving the above-formalized construction planning problem. Based on the analysis of the difficulties during the planning, the overall framework of the planning model is first presented, and then two highlights of the model are explained in detailed.

3.1. Overall framework of the planning model

Based on the classical HTN planning process, the overall task *cgTask* will be recursively decomposed into primitive tasks that can be completed by instantiating construction activities. It is a state-based forward search process that relies on the state transition that corresponds to the state change after applying construction activities to primitive tasks. Moreover, the earlier

applied activities are considered to be executed first. In this way, activities in the generated action plans are totally ordered, without the concurrency of activities into consideration. Obviously, by allowing concurrent executions of construction activities, it is able to shorten the project duration. In order to satisfy the deadline constraint, it is required to extend the planning process and enable it to deal with the concurrency of construction activities.

On the other hand, the HTN planning performs depth-first search in the refinement of abstract tasks, which means it always proceeds down a branch of the hierarchical task network to decompose an abstract task until reaching the bottom or encountering an infeasible branch due to the violations of time and cost constraints. In this case, the planning process has to backtrack to try other branches. During the construction planning, it is very likely that the selected construction method for earlier applied construction activities result in the subsequent search being unable to continue because the cost of the current plan exceeds the budget or the duration of the plan exceeds the deadline. When this kind of violation happens at a later time, it will cause a lot of backtrack, which is time consuming.

To address the above-mentioned deficiencies of classical HTN planning with respect to construction planning, a construction planning model based on HTN (abbreviated as CP-HTN) is proposed to realize the desired planning process. The overall framework of CP-HTN is shown in Figure 3. The planning model takes as input the domain-dependent knowledge, the project-related information and the overall task, and outputs a construction plan. Specifically, abstract tasks in the task network are decomposed into subtasks according to the task decomposition knowledge, and in turn the task network is updated by replacing the decomposed abstract task of its subtasks. On the other hand, primitive tasks are solved by applying the corresponding construction activities, followed by the deletion of the primitive tasks from the task network. This process is repeated until the task network is empty, then a construction plan is obtained. During the process of applying construction activities, tem-

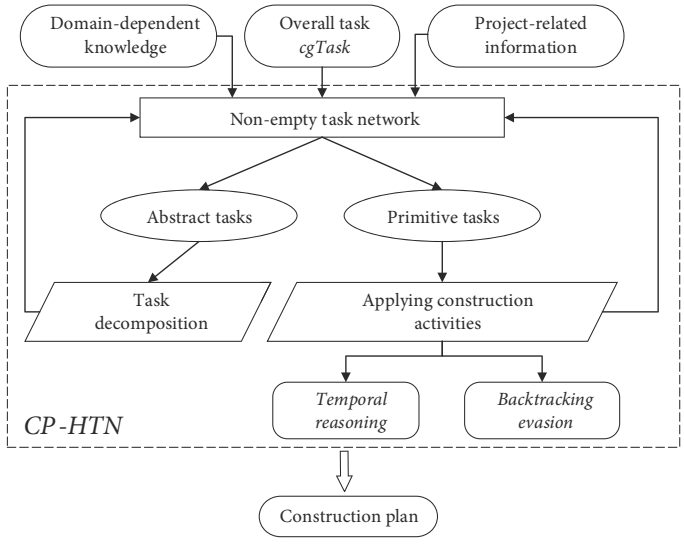

Figure 3. Overall framework of CP-HTN model

poral reasoning technique is adopted to enable the planning model to deal with the concurrent executions of construction activities, and backtracking evasion mechanism is devised to eliminate unnecessary backtracking. These two highlights are explained briefly as follows, and more details are described later.

(1) The temporal reasoning technique only generates necessary precedence relations between construction activities to guarantee the obtained construction plan is viable with respect to the interdependence between construction components, as a result, concurrent execution of construction activities that corresponds to components without dependencies is allowed.

(2) The backtracking evasion mechanism postpones the instantiation of numeric parameters until it must be done. In this way, backtracking caused by inappropriate instantiation of numeric parameters at an early time can be avoided. To ensure the final construction plan satisfies the time and cost constraints, feasibility checking is performed to find feasible solution for the numeric parameters with respect to deadline and budget limit before they are instantiated.

3.2. Temporal reasoning

In construction planning, it is important to correctly sequence the construction activities so that these activities can be executed without unnecessary rework due to conflicts with construction activities already executed. At the same time, it is expected that construction activities without sequential relationships can be executed concurrently. However, the traditional HTN planning assumes that the order in which construction activities are applied is the order of their execution, which will add unnecessary sequential constraints between construction activities and make the concurrent execution of construction activities impossible. To avoid the deficiency, the planning model should only generate necessary sequential constraints between construction activities to prevent conflicts from taking place. Since sequencing knowledge cannot be explicitly expressed in the domain-dependent knowledge, sequential relations are derived from the interaction between the execution of construction activities. From this perspective, two cases will result in conflicts if construction activities are arranged to be executed in a wrong order.

(1) **Conflict 1:** Assuming that the effect of construction activity A adds the literals to the construction state, which are required in the precondition of construction activity B, then a conflict will occur if B is arranged to execute before A;

(2) **Conflict 2:** Assuming that the effect of construction activity A deletes the literals from the construction state, which are required in the precondition of construction activity B, then a conflict will occur if A is arranged to execute before B.

For the purpose of avoiding infeasible activities, sequence constraints between construction activities are enforced to eliminate the possible conflicts, which are derived from the possible interactions between preconditions and effects of different construction activities. The derivation process is referred to as temporal reasoning, which conforms to the following rules.

(1) **Rule 1:** If the precondition of construction activity *CA-A* requires the support of the *add* effect of construction activity *CA-B*, then *CA-A* is supposed to start after *CA-B*

ends. For example, construction activity for beams should start after the supporting columns are completed.

(2) **Rule 2:** If the *del* effect construction activity *CA-A* invalidates the precondition of construction activity *CA-B*, then *CA-A* is supposed to start after *CA-B* ends. For example, construction activity for dismantling scaffolds should start after the completion of masonry wall which requires scaffolding.

According to these two rules, temporal constraints will be generated when applying a construction activity to ensure no conflict will take place. Based on these constraints, it is possible to schedule the applied construction activities until now to check whether the deadline is met. Guaranteed by the checking, the final construction plan is supposed to meet the requirement of project duration.

3.3. Mechanism for backtracking evasion

Due to the different construction technologies, there are more than one execution mode to complete a construction activity, and the time and cost required for different execution modes are different. Under the limitations of deadline and budget, the selection of execution mode for different construction actions interact with each other. For example, when execution modes with high cost but low duration are chosen for construction activities applied early, subsequent construction activities can only be executed in a mode with low cost but high duration in order to not exceed the deadline and budget. Moreover, the same type of components appear in different locations, causing the identical construction activities to be performed at different times, and these activities must be completed in the same execution mode. In this case, if an inappropriate execution mode is chosen for a construction activity when it is applied for the first time, unsatisfactory time or budget may occur when such construction activities are applied again. This will result in that the planning process backtracks to the state when the construction is first applied and chooses another execution mode to proceed.

To make the above description clear, an example is taken here. Assume that the current partial construction plan consists of three construction activities that are totally ordered, i.e. $P = \{A_1 \rightarrow A_2 \rightarrow A_3\}$. As the planning process continues, new construction activities are applied and added into the partial plan. When a construction activity of the same type of A1 is applied to form a new partial plan $P' = \{A_1 \rightarrow A_2 \rightarrow A_3 \rightarrow \ldots \rightarrow A_n\}$, the duration of the new partial plan exceeds the deadline because the time required to execution this activity is too long. Since the execution modes with the smallest viable duration have been selected for construction activities A_2 to A_{n-1} have, it is possible to make the partial plan temporally viable only by selecting an execution mode with smaller duration for construction activities A_1 and A_n. This means the planning process must backtrack to the state when A_1 is applied, which is a time-consuming process.

In order to avoid the above-mentioned backtracking, a backtracking evasion mechanism is developed to postpone the selection of execution mode for each activity until the end. Specifically, each time a construction activity is applied by instantiating the construction activity template, its execution mode is not instantiated but recorded as a variable. To ex-

amine whether the newly applied activity is valid, it is necessary to check whether there exists a combination of execution modes for applied activities such that the current partial construction plan satisfies the deadline and budget constraints. If such a valid combination exists, the newly applied activities is added to the partial construction plan. Otherwise, the planning process will withdraw this activity and backtrack to last decision point where an abstract task or a primitive task is selected to solve.

The validity checking of the newly applied activities is modelled as a constraints satisfaction problem (CSP). Concretely, assume that the current partial construction plan is composed of a set of construction activities, denoted as A, and a set of temporal constraints, denoted as SR. With the deadline and budget limits, the problem is formulated as follows.

(1) Decision variables: The M decision variables relate to the selection of an appropriate execution mode for each type of construction activity included in the partial construction plan, i.e.

$$M_{ik} = \{0,1\},\ \forall k = 1,\ldots,\ K_i,$$

where M_{ik} is a binary decision variable to select an execution mode k for construction activities of type i, each execution mode defines the activity duration and cost. By selecting an execution mode for construction activities of type i, its duration (d_{ik}) and cost (c_{ik}) are directly determined. Thus, the duration (D_i) and cost (C_i) of construction activities of type i can be expressed as a function of which execution mode is selected, as follows:

$$D_i = \sum_{k=1}^{K_i} d_{ik} M_{ik};$$

$$C_i = \sum_{k=1}^{K_i} c_{ik} M_{ik}.$$

(2) Constraint on cost: The constraint defines the limitation on the total cost of the partial construction plan, which can be expressed by the following inequality.

$$\sum_{j=1}^{|A|} \sum_{i=1}^{K_i} c_{ik} M_{ik} \le B,$$

where |A| denotes the number of construction activities in the partial construction plan, i is the activity type, and B is the specified budget.

(3) Constraints on sequence relations: These constraints correspond to the temporal constraints derived from applying construction activities. Each temporal constraint in SR has the form of <i, j> which means that activity i precedes activity j, and can be expressed as an inequality of scheduled start time and duration.

$$SS_i + \sum_{k=1}^{K_m} d_{mk} M_{mk} \le SS_j,$$

where SS_i and SS_j are respectively the scheduled start time of activity i and activity j, m represents the type of activity i.

(4) Constraint on deadline: This constraint restrict the project duration to not exceed the deadline (D), which means the latest scheduled finish time of all the activities should come before the deadline, as follows:

$$\max_{i \in A} SS_i + \sum_{k=1}^{K_m} d_{mk} M_{mk} \leq D.$$

Solving this CSP is to find a set of feasible assignments for decision variables such that the above constraints are all satisfied. In artificial intelligence, backtracking search (BS) is a complete and practical algorithm to solve CSP (Van Beek, 2006), therefore, it is adopted here to search for the solution that corresponds to a feasible combination of execution modes for construction activities in the current partial plan. It is worth noting that backtracking of BS focuses on the infeasible execution mode of a certain type of construction activities, differing from that of CP-HTN planning which focuses on the invalid construction activities. As will be seen in the next section, backtracking on infeasible execution mode is more efficient.

According to the above idea and the temporal reasoning technique, a procedure is designed to apply the construction activities in the planning process, shown in Table 2. To make it easier to understand the procedure of instantiating the construction activity template, some data structures used in the procedure are explained as below.

(1) *index-actions* is a set of triples that record information of the applied construction activities, each of which has the form of (*step, amount, type*) where *step* is a number used to index the construction activity, *amount* is its engineering quantities, and *type* is used to differentiate between different types of activities which are associated with different set of alternative execution modes.

(2) $L^+(l)/L^-(l)$ gives the index of construction activity by which literal l of construction state was most recently added/deleted.

Table 2. Procedure for defining construction activity

Legend: p-index is a global variable for indexing the construction activity, initially equals to zero
Procedure define-CA (CS_0, *CA, Modes*) **1 if** CS_0 does not satisfy *CA.pre* **then** backtrack to task decomposition **2** *step* ← *p-index* + 1 **3** add (*step, CA.ExM.amount, CA.ExM.mode*) to *index-actions* **4 for** $l \in$ *CA.pre* **5** **if** $L^+(l)$ exists **then** add ($L^+(l)$, *step*) to *SR* **6 for** $l \in$ *CA.eff.del* **7** **if** $L^+(l)$ exists **then a**dd ($L^+(l)$, *step*) to *SR* **8** $L^-(l) \leftarrow$ *step* **9 for** $l \in$ *CA.eff.add* **10** $L^+(l) \leftarrow$ *step* **11** extract activity types *CATypes* ← {*type* | *type* exists in *index-actions*} **12** search for a feasible combination of execution modes Sol = BS (*CATypes, index-actions, SR, modes*) **13 if** Sol does not exist **then** backtrack to task decomposition **14 return** $s' \leftarrow (s - a.del_s) \cup a.add_s$

(3) *SR* is a set of temporal constraints between construction activities, each of which has the form of (*step1 step2*) indicating that construction activity indexed by *step1* comes before construction activity indexed by *step2*.

(4) *Modes* is a set of execution modes for all types of construction activities, each of which is a set of available execution modes for a certain type of construction activity, denoted by *CA-modes*. Each execution mode in *CA-modes* is represented as (*mName, pr, cpd*) in which *mName* represents the mode name, such as artificial excavation, mechanic excavation, etc.; *pr* is the production rate and *cpd* is the unit cost per day. Thus, the duration (*d*) and cost (*c*) for completing the construction activity in the specified execution mode can be estimated by the following formulas: $d = Q/pr$ and $c = d*cpd$, where Q represents the engineering quantities.

Since the above procedure can prevent invalid construction activity from being added to the construction plan, a solution returned by the planning process is bound to meet the deadline and budget requirements. The solution is a set of instantiated construction activities, associated with the recorded information and sequence constraints. To get the executable construction plan, backtracking search is invoked again to find a feasible combination of execution modes for each construction activity.

4. Experimental studies

To demonstrate the feasibility and practicability of the proposed planning model, a set of experimental studies involving two comparisons is carried out in this section. One is to compare the construction plans for the same construction problems but with a different deadline and budget limits, while the other is to compare the proposed planning model with similar planning approach.

The case study is related to complete the main structure of a three-story building within specific deadline and budget. The hierarchical product model of the involved building is shown in Figure 4, and the topological relationships between primitive construction objects

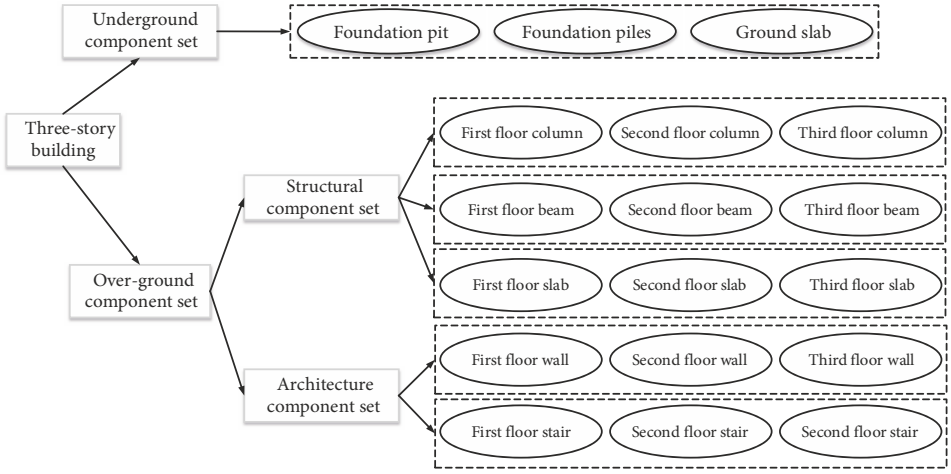

Figure 4. Hierarchical product model of a three-story building

are given in Table 3. In addition, to support the estimation of duration and cost of the construction activities, the engineering quantities of each primitive construction project are also provided in Table 3, and the production rate and unit cost for different construction methods are listed in Table 4.

Table 3. Information of primitive construction objects

Construction object	Identifier	Quantities	Relationship
foundation pit	founda_pit	1256.6 m^3	–
foundation pile	founda_pile	188.4 m^3	(embedded-in founda_piles founda_pit)
ground slab	ground_slab	73.8 m^3	(supported-by ground_slab founda_pile)
first floor column	1F_column	44.71 m^3	(supported-by 1F_column ground_slab)
first floor beam	1F_beam	59.37 m^3	(supported-by 1F_beam 1F_column)
first floor slab	1F_slab	73.85 m^3	(supported-by 1F_slab 1F_beam)
second floor column	2F_column	31.23 m^3	(supported-by 2F_cloumn 1F_slab)
second floor beam	2F_beam	53.06 m^3	(supported-by 2F_beam 2F_column)
second floor slab	2F_slab	80.49 m^3	(supported-by 2F_slab 2F_beam)
third floor column	3F_column	30.48 m^3	(supported-by 3F_cloumn 2F_slab)
third floor beam	3F_beam	53.75 m^3	(supported-by 3F_beam 3F_column)
third floor slab	3F_slab	88.42 m^3	(supported-by 3F_slab 3F_beam)
first floor wall	1F_wall	153.6 m^2	(enclosed_by 1F_wall ground_slab 1F_column 1F_slab)
first floor stair	1F_stair	16.08 m^3	(connect 1F_stair ground_slab 1F_slab)
second floor wall	2F_wall	138.2 m^2	(enclosed_by 2F_wall 1F_slab 2F_column 2F_slab)
second floor stair	2F_stair	13.12 m^3	(connect 2F_stair 1F_slab 2F_slab)
third floor wall	3F_wall	138.2 m^2	(enclosed_by 3F_wall 2F_slab 3F_column 3F_slab)
third floor stair	3F_stair	13.02 m^3	(connect 3F_stair 2F_slab 3F_slab)

Table 4. The production rate and unit cost of construction method

No.	Construction task type	Construction method	Production rate	Unit cost
1	Foundation-pit-excavation	Artificial excavation	98.6 m^3/day	5000 CNY/day
		Mechanical excavation	259.3 m^3/day	22000 CNY/day
2	Earthwork-backfilling	Manual tamping	128.4 m^3/day	2000 CNY/day
		Mechanical compaction	249.2 m^3/day	4000 CNY/day
3	Foundation-piles-construction	Cast-in-place concrete piles	28.6 m^3/day	20000 CNY/day
		Prefabricated piles	42.6 m^3/day	35000 CNY/day
4	Reinforced-concrete-engineering	Cast-in-situ	18.36 m^3/day	12000 CNY/day
		Precast	46.5 m^3/day	36000 CNY/day
5	Masonry-wall	Trinity bricklaying	45.2 m^2/day	4000 CNY/day
		Shove joint brickwork	76.4 m^2/day	5000 CNY/day

In addition, the geological and hydrological conditions of the construction site affect the selection of construction method for underground components. For the three-story building, it is assumed that the depth of the foundation pit is less than the groundwater level, which means that no measure for reducing groundwater level is needed. Also, it is assumed that the compaction degree of foundation soil is high and thus suitable to serve as a natural foundation.

4.1. Comparison between construction plans for scenarios with different deadline and budget

To demonstrate the feasibility of the proposed planning model, an experimental study that compares the construction plans for two scenarios with a different deadline and budget limits is carried out. Scenario 1 has the loose deadline but the tight budget, while scenario 2 has the tight deadline but loose budget. According to the provided data, the maximum and minimum of the project duration are respectively 61 days and 29 days, and the same bounds of the total cost are 1,064,000 CNY and 740,000 CNY. On this basis, the duration and budget of scenario-1 are set to 50 days and 900,000 CNY, while those of scenario-2 are set to 35 days and 1,050,000 CNY.

To get feasible construction plans for two scenarios, the information in Table 2 and Table 3, as well as the relevant construction planning domain knowledge, are encoded and input to the planning system tailored on the basis of SHOP2 (Nau et al., 2003), which is the most used open source HTN planner. According to the generated construction plans, two Gantt charts are drawn to illustrate the sequential relationships between construction activities, as well as the their durations reflected by the length of the bar blocks, shown in Figure 5 and Figure 6. Comparing these two figures, it can be found that the same number of construction activities are defined to complete the construction of all the components, and the activities sequencing of construction activities are the same because of the identical topological relationships between components. The only difference is reflected in the selection of execution modes, as shown in Table 5.

Recalling the duration and cost of different execution modes, it is not difficult to understand the difference in mode selection in both scenarios. To meet the requirement of loose deadline and tight budget in scenario 1, it was natural to select the execution mode with less cost for construction activities. Since the reinforced concrete task occupies most of the engineering volume, it was inevitable to select the "cast-in-situ" mode for all construction activities responsible for reinforced concrete components. At the same time, the "Shove joint brickwork" mode and the "Prefabricated" mode were selected for masonry wall task and foundation piles task respectively in order to shorten the total duration with less cost increase. In contrast, "Precast" mode with smaller duration was selected for reinforced concrete components in scenario 2 to satisfy the deadline limit. On the other hand, execution modes with less cost are selected for foundation piles task and masonry wall task without increasing project duration much. From this comparison, it can be seen that the deadline and budget limits do affect the selection of execution modes, and the proposed planning model is able to find a feasible solution under the limitations.

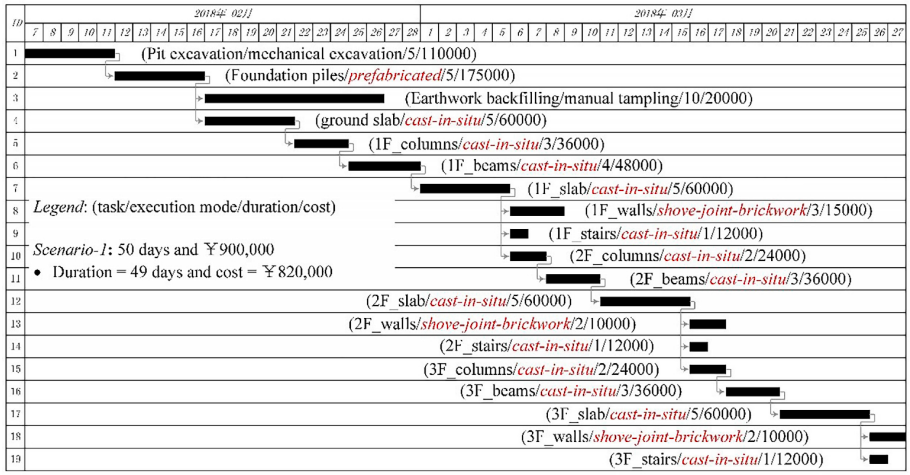

Figure 5. A feasible construction Plan for scenario 1

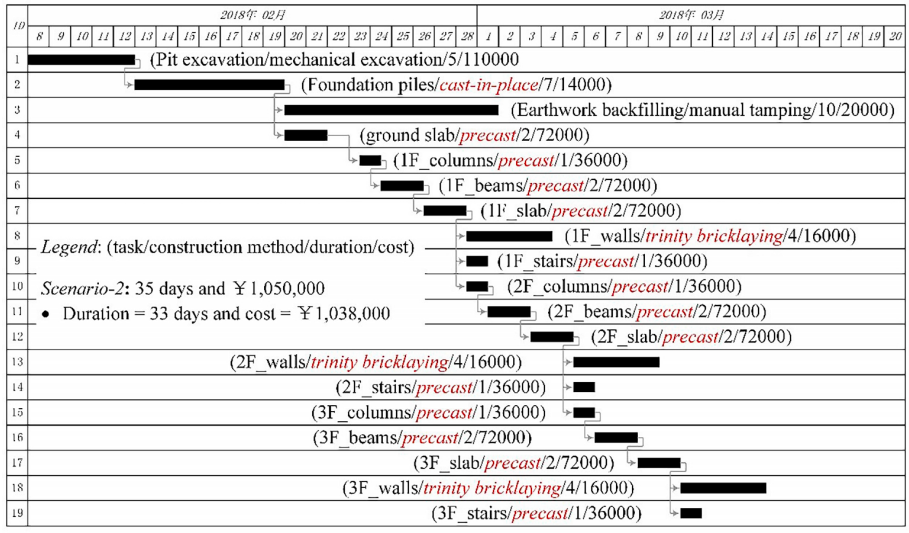

Figure 6. A feasible construction plan for scenario 2

Table 5. Selection of execution modes in two scenarios

Construction task type	Execution mode	
	Scenario 1	Scenario 2
Foundation-pit-excavation	Mechanical excavation	Mechanical excavation
Earthwork-backfilling	Manual tamping	Manual tamping
Foundation-piles-construction	Prefabricated	Cast-in-place
Reinforced-concrete-engineering	Cast-in-situ	Precast
Masonry-wall	Shove joint brickwork	Trinity bricklaying

4.2. Comparison between CP-HTN and SHOP2

In order to justify that the backtracking evasion mechanism improves the performance of HTN planning on solving the construction planning problem, another experimental study is performed to compare the performance of CP-HTN and classic HTN planner SHOP2. Based on the hypothetical construction project, thirteen problems are designed to test those two planning approaches, the information of which is listed in Table 6. The first three problems have different numbers of construction objects, and thus need different numbers of construction activities. In the middle five problems, the execution modes requiring less duration but more cost for construction tasks No. 1 to No. 5 are deleted one by one, while in the last five problems, construction objects on the third and second floor are deleted and the execution modes with more duration but less cost for construction tasks No. 1 to No. 5 are deleted one by one. Each problem also has two scenarios, the deadline and budget requirements of which are determined by the following way: selecting the first execution modes for construction tasks with two alternatives in scenario 1 while selecting the second execution modes in scenario 2.

Table 6. Information of testing problems

Problems	Description	Scenario 1	Scenario 2
Problem 1	is identical to the problem used in the first comparison	61 d/￥753,000	29 d/￥1,064,000
Problem 2	Construction objects on third floor are deleted	51 d/￥665,000	28 d/￥838,000
Problem 3	Construction objects on the second and third floor are deleted	41 d/￥457,000	20 d/￥612,000
Problem 4	Compared to Problem 1, second execution modes for construction task No. 1 is deleted	61 d/￥753,000	37 d/￥1,019,000
Problem 5	Compared to Problem 1, second execution modes for construction tasks No. 1 to No. 2 are deleted	61 d/￥753,000	37 d/￥1,015,000
Problem 6	Compared to Problem 1, second execution modes for construction tasks No. 1 to No. 3 are deleted	61 d/￥753,000	39 d/￥980,000
Problem 7	Compared to Problem 1, second execution modes for construction tasks No. 1 to No. 4 are deleted	61 d/￥753,000	59 d/￥740,000
Problem 8	Compared to Problem 1, second execution modes for construction tasks No. 1 to No. 5 are deleted	61 d/￥753,000	61 d/￥753,000
Problem 9	Compared to Problem 3, first execution mode for construction task No. 1 is deleted	33 d/￥502,000	20 d/￥612,000
Problem 10	Compared to Problem 3, first execution modes for construction tasks No. 1 and No. 2 are deleted	33 d/￥506,000	20 d/￥612,000
Problem 11	Compared to Problem 3, first execution modes for construction tasks No.1 to No. 3 are deleted	31 d/￥542,000	20 d/￥612,000
Problem 12	Compared to Problem 3, first execution modes for construction tasks No.1 to No. 4 are deleted	21 d/￥613,000	20 d/￥612,000
Problem 13	Compared to Problem 3, first execution modes for construction tasks No. 1 to No. 5 are deleted	20 d/￥612,000	20 d/￥612,000

To make a fair comparison, the same temporal reasoning technique is integrated into SHOP2 to deal with deadline limits because SHOP2 has limited ability to infer temporal constraints. CP-HTN and SHOP2 are implemented with Standard Bank Common Lisp (SBCL) on Emacs-25.3 and tested on each problem ten times on an Intel(R) Xeon® CPU W3530 @ 2.80GHz with 4G RAM. CUP time is used to show their performance, and the final results are shown in Table 7.

Table 7. Comparison results of CP-HTN and SHOP2

Problems	CP-HTN-CPU time (s)		SHOP2-CPU time (s)	
	Scenario 1	Scenario 2	Scenario 1	Scenario 2
Problem 1	0.016	0.016	0.016	>5
Problem 2	0.010	0.010	0.011	>5
Problem 3	0.006	0.006	0.005	>5
Problem 4	0.016	0.015	0.016	>5
Problem 5	0.015	0.016	0.015	>5
Problem 6	0.017	0.016	0.016	>5
Problem 7	0.016	0.016	0.015	>5
Problem 8	0.016	0.016	0.015	0.015
Problem 9	0.006	0.006	0.004	>5
Problem 10	0.005	0.006	0.005	>5
Problem 11	0.005	0.006	0.005	>5
Problem 12	0.006	0.006	0.005	0.140
Problem 13	0.006	0.006	0.004	0.004

The results show that CP-HTN is able to solve all the problems in two scenarios, but SHOP2 can only find a feasible solution for all problems in scenario 1 and few problems in scenario 2. With regard to Problems 1 to 3, the planning time for scenario 1 decreases as the number of construction activities decreases, but the planning time for all three problems exceeds the specified time limit. It indicates that the reduction in the number of construction activities does not improve the time increase caused by backtracking. As to Problems 4 to 7, the planning time was still timed out although the second execution modes were constantly deleted, and a feasible solution is obtained for Problem 8 within the specified time until the execution modes for each type of construction tasks were determined. When it comes to Problems 9–13, the similar phenomena occurs except that a feasible solution was found quickly for Problem 12, which can be attributed to the fact that the masonry wall task was only applied at the end while being applied in different planning depths in Problem 7.

To figure out the reason behind these phenomenon, we step into the planning process of SHOP2 and find that, guided by the depth-first search mechanism, SHOP2 always choose the first execution mode for a type of construction tasks. In scenario 1, this approach happens to find the solution quickly, but in scenario 2, each time the first execution mode is chosen

for a construction task will result in backtracking and the backtracking will grow exponentially with the planning depth. According to the setting of scenario 2, its feasible solution is the combination of the second execution modes for each type of construction task with two alternative modes. Each time the planning process selected the first execution mode, the obtained partial plan would be infeasible and the planning process must withdrew the selection and tried the second execution mode. This kind of backtracking would occur each time when a construction activity with two alternative execution modes was applied. The later the construction activity was applied, the longer time the backtracking would spend. This is why SHOP2 can find a solution quickly for Problem 12 in scenario 2 but cannot find a solution for Problem 7 in scenario 2 within a specified time.

By comparison, the success of CP-HTN justifies that it does not suffer from the backtracking caused by the inappropriate selection of execution modes for construction tasks in different planning depths. This is because the developed backtracking evasion mechanism does not determine the execution mode when defining a construction activity, but determine the appropriate combination of construction methods after all the construction activities are defined while ensuring the defined activities are valid. Meanwhile, the backtracking evasion mechanism has similar performance with SHOP2 on all problems in scenario 1. This also shows that CP-HTN needs to check the validity of the defined activities, but it does not affect its efficiency. In summary, the backtracking evasion mechanism makes CP-HTN being an effective tool to solve construction planning problem with a deadline and budget limits.

Conclusions

This paper presents a construction planning model for time and cost-constrained construction project on the basis of HTN planning. The planning model employed temporal reasoning technique to derive sequential relationships between construction activities, which were used to examine whether the obtained partial/complete construction plan satisfied the deadline constraint. On the other hand, a backtracking evasion mechanism was developed to free the planning model from the trouble of backtracking encountered by the traditional HTN planning. The comparison between construction plans for two scenario demonstrated that the time and cost constraints did indeed affect the selection of execution modes for construction activities. Meanwhile, the comparison between CP-HTN and SHOP2 justified that the proposed backtracking evasion mechanism was able to avoid the trouble of backtracking when selecting valid execution modes for different construction activities such that the plan duration didn't exceed the deadline and the total cost was within the specified budget. Although the experiment is conducted on a hypothetical project, the proposed approach is suitable for practical construction projects because HTN planning is adequate to handle large-scale real-word planning problems. Thus, in the future, experiments will be conducted on real construction projects to demonstrate the feasibility of the practice. On the other hand, uncertainty is ubiquitous in the practical construction project, which will result in uncertain durations. In this context, how to deal with those uncertainties and develop deadline-satisfied construction plan needs further research efforts.

Acknowledgements

Thanks are due to the referees for their valuable comments.

Funding

This work was supported by the National Natural Science Foundation of China under Grant [No. 71390524, No. 71821001]; and the Research Grants Council of Hong Kong under Grant [PolyU 152093/14E].

Disclosure statement

No competing financial, professional, or personal interests exist.

References

Abuwarda, Z., & Hegazy, T. (2016). Work-package planning and schedule optimization for projects with evolving constraints. *Journal of Computing in Civil Engineering*, *30*(6), 04016022. https://doi.org/10.1061/(ASCE)CP.1943-5487.0000587

An, S.-M., Woo, S., Cho, C.-S., & Lee, S. (2017). Development of budget-constrained rescheduling method in mega construction project. *KSCE Journal of Civil Engineering*, *21*(1), 85-93. https://doi.org/10.1007/s12205-016-0966-7

Benjamin, C. O., Babcock, D. L., Yunus, N. B., & Kincaid, J. (1990). Knowledge-based prototype for improving scheduling productivity. *Journal of Computing in Civil Engineering*, *4*(2), 124-134. https://doi.org/10.1061/(ASCE)0887-3801(1990)4:2(124)

Dzeng, R.-J., & Tommelein, I. D. (1997). Boiler erection scheduling using product models and case-based reasoning. *Journal of Construction Engineering and Management*, *123*(3), 338-347. https://doi.org/10.1061/(ASCE)0733-9364(1997)123:3(338)

Dzeng, R.-J., & Tommelein, I. D. (2004). Product modeling to support case-based construction planning and scheduling. *Automation in Construction*, *13*(3), 341-360. https://doi.org/10.1016/j.autcon.2003.10.002

Echeverry, D., Ibbs, C. W., & Kim, S. (1991). Sequencing knowledge for construction scheduling. *Journal of Construction Engineering and Management*, *117*(1), 118-130. https://doi.org/10.1061/(ASCE)0733-9364(1991)117:1(118)

Faghihi, V., Nejat, A., Reinschmidt, K. F., & Kang, J. H. (2015). Automation in construction scheduling: a review of the literature. *The International Journal of Advanced Manufacturing Technology*, *81*(9-12), 1845-1856. https://doi.org/10.1007/s00170-015-7339-0

Fischer, M. A., & Aalami, F. (1996). Scheduling with computer-interpretable construction method models. *Journal of Construction Engineering and Management*, *122*(4), 337-347. https://doi.org/10.1061/(ASCE)0733-9364(1996)122:4(337)

Ghallab, M., Nau, D., & Traverso, P. (2004). Chapter 11 - hierarchical task network planning. In M. Ghallab, D. Nau, & P. Traverso (Eds.), *Automated Planning* (pp. 229-261). Burlington: Morgan Kaufmann.

Hendrickson, C., & Au, T. (1989). *Project management for construction: Fundamental concepts for owners, engineers, architects, and builders*. Englewood Cliffs: Prentice-Hall.

Hendrickson, C., Zozaya-Gorostiza, C., Rehak, D., Baracco-Miller, E., & Lim, P. (1987). Expert system for construction planning. *Journal of Computing in Civil Engineering*, *1*(4), 253-269. https://doi.org/10.1061/(ASCE)0887-3801(1987)1:4(253)

Hu, X., Xia, B., Skitmore, M., & Chen, Q. (2016). The application of case-based reasoning in construction management research: An overview. *Automation in Construction*, *72*, 65-74. https://doi.org/10.1016/j.autcon.2016.08.023

Jägbeck, A. (1994). MDA planner: interactive planning tool using product models and construction methods. *Journal of Computing in Civil Engineering*, *8*(4), 536-554. https://doi.org/10.1061/(ASCE)0887-3801(1994)8:4(536)

Kannimuthu, M., Ekambaram, P., Raphael, B., & Kuppuswamy, A. (2018). Resource unconstrained and constrained project scheduling problems and practices in a multiproject environment. *Advances in Civil Engineering*, 1-13. https://doi.org/10.1155/2018/9579273

Kartam, N. A., & Levitt, R. E. (1990). Intelligent planning of construction projects. *Journal of Computing in Civil Engineering*, *4*(2), 155-176. https://doi.org/10.1061/(ASCE)0887-3801(1990)4:2(155)

Kartam, N. A., Levitt, R. E., & Wilkins, D. E. (1991). Extending artificial intelligence techniques for hierarchical planning. *Journal of Computing in Civil Engineering*, *5*(4), 464-477. https://doi.org/10.1061/(ASCE)0887-3801(1991)5:4(464)

Lee, K. J., Kim, H. W., Lee, J. K., & Kim, T. H. (1998). Case-and constraint-based project planning for apartment construction. *AI Magazine*, *19*(1), 13-24. https://doi.org/10.1609/aimag.v19i1.1350

Liu, D., Wang, H., Qi, C., Zhao, P., & Wang, J. (2016). Hierarchical task network-based emergency task planning with incomplete information, concurrency and uncertain duration. *Knowledge-Based Systems*, *112*, 67-79. https://doi.org/10.1016/j.knosys.2016.08.029

Nau, D. S. (2007, December 15). Current trends in automated planning. *AI Magazine*, *28*, 43-58. https://doi.org/10.1609/aimag.v28i4.2067

Nau, D. S., Au, T.-C., Ilghami, O., Kuter, U., Murdock, J. W., Wu, D., & Yaman, F. (2003). SHOP2: An HTN planning system. *Journal of Artificial Intelligence Research*, *20*, 379-404. https://doi.org/10.1613/jair.1141

Olawale, Y. A., & Sun, M. (2010). Cost and time control of construction projects: inhibiting factors and mitigating measures in practice. *Construction Management and Economics*, *28*(5), 509-526. https://doi.org/10.1080/01446191003674519

Qi, C., Wang, D., Muñoz-Avila, H., Zhao, P., & Wang, H. (2017). Hierarchical task network planning with resources and temporal constraints. *Knowledge-Based Systems*, *133*, 17-32. https://doi.org/10.1016/j.knosys.2017.06.036

Ryu, H.-G., Lee, H.-S., & Park, M. (2007). Construction planning method using case-based reasoning (CONPLA-CBR). *Journal of Computing in Civil Engineering*, *21*(6), 410-422. https://doi.org/10.1061/(ASCE)0887-3801(2007)21:6(410)

Shaked, O., & Warszawski, A. (1995). Knowledge-based system for construction planning of high-rise buildings. *Journal of Construction Engineering and Management*, *121*(2), 172-182. https://doi.org/10.1061/(ASCE)0733-9364(1995)121:2(172)

Tah, J. H. ., Carr, V., & Howes, R. (1999). Information modelling for case-based construction planning of highway bridge projects. *Advances in Engineering Software*, *30*(7), 495-509. https://doi.org/10.1016/S0965-9978(98)00128-8

Tah, J. H. M., Carr, V., & Howes, R. (1998). An application of case-based reasoning to the planning of highway bridge construction. *Engineering Construction and Architectural Management*, *5*(4), 327-338. https://doi.org/10.1046/j.1365-232X.1998.54069.x

Urizar, M., & Halim, E.-S. A. (2015). *Construction supervision QC + HSE management in practice : quality control, OHS, and environmental performance reference guide*. Gordon NSW: Xlibris AU.

Van Beek, P. (2006). Backtracking search algorithms. *Foundations of Artificial Intelligence*, *2*, 85-134. https://doi.org/10.1016/S1574-6526(06)80008-8

Vanhoucke, M. (2018). Planning projects with scarce resources: Yesterday, today and tomorrow's research challenge. *Frontiers of Engineering Management*, *5*(2), 133-149. https://doi.org/10.15302/J-FEM-2018088

Xu, K., & Muñoz-Avila, H. (2008). CaBMA: A case-based reasoning system for capturing, refining, and reusing project plans. *Knowledge and Information Systems*, *15*(2), 215-232. https://doi.org/10.1007/s10115-007-0077-3

Zhang, Y., Ding, L., & Love, P. E. D. (2017). Planning of deep foundation construction technical specifications using improved case-based reasoning with weighted k-nearest neighbors. *Journal of Computing in Civil Engineering*, *31*(5), 04017029. https://doi.org/10.1061/(ASCE)CP.1943-5487.0000682

Zhou, J., Love, P. E. D., Wang, X., Teo, K. L., & Irani, Z. (2013). A review of methods and algorithms for optimizing construction scheduling. *Journal of the Operational Research Society*, *64*(8), 1091-1105. https://doi.org/10.1057/jors.2012.174

Zozaya-Gorostiza, C., Hendrickson, C., & Rehak, D. R. (1989). *Knowledge-based process planning for construction and manufacturing. Knowledge-based process planning for construction and manufacturing*. San Diego, CA, USA: Academic Press. https://doi.org/10.1016/B978-0-12-781900-6.X5001-7