QUN CHEN, Ph.D.
E-mail: chenqun631@mail.csu.edu.cn
Central South University
School of Traffic and Transportation Engineering
CSU Railway Campus, Changsha, 410075, China
HAIBO CHEN, Ph.D.
E-mail: H.Chen@its.leeds.ac.uk
University of Leeds
Institute for Transport Studies
Leeds LS2 9JT, United Kingdom

Science in Traffic and Transportation
Original Scientific Paper
Accepted: Apr. 3, 2012
Approved: Sep. 24, 2013

SOLUTION ALGORITHM FOR A NEW BI-LEVEL DISCRETE NETWORK DESIGN PROBLEM

ABSTRACT

A new discrete network design problem (DNDP) was proposed in this paper, where the variables can be a series of integers rather than just 0-1. The new DNDP can determine both capacity improvement grades of reconstruction roads and locations and capacity grades of newly added roads, and thus complies with the practical projects where road capacity can only be some discrete levels corresponding to the number of lanes of roads. This paper designed a solution algorithm combining branch-and-bound with Hooke-Jeeves algorithm, where feasible integer solutions are recorded in searching the process of Hooke-Jeeves algorithm, lending itself to determine the upper bound of the upper-level problem. The thresholds for branch cutting and ending were set for earlier convergence. Numerical examples are given to demonstrate the efficiency of the proposed algorithm.

KEY WORDS

discrete network design problem, integer programming, branch-and-bound algorithm

1. INTRODUCTION

The network design problem (NDP) is concerned with the modification of a transportation network configuration by adding new links or improving the existing ones, so that the total travel time over the network or the total cost including travel time and investment is minimized as the certain social welfare objective. Selecting the locations of the additional links and determining their capacities are motivating problems, trying to minimize the total system costs and accounting for the route choice behaviour of network users. NDP can be roughly classified into three categories. The standard DNDP, expressed by 0-1 integer decision variables, deals with the selection of the optimal locations of new links to be added. The continuous network design problem (CNDP), expressed by continuous decision variables, determines the optimal capacity enhancement for a subset of existing links. Finally, the mixed network design problem (MNDP) combines both CNDP and DNDP in a network [1]. NDP can be generally formulated as mathematical programming with equilibrium constraints (MPEC), determinate user equilibrium assignment model (UE) or stochastic user equilibrium assignment model (SUE), mentioned in [2, 3, 4]. MPEC is usually applied to describe the route choice behaviour of network users. Various solution algorithms, including gradient-based [5, 6, 7, 8] and derivative-free (or meta-) heuristic algorithms [9, 10, 11], have been proposed for solving the family of NDPs.

Because of the computational difficulties experienced with the solution algorithm of non-linear bi-level mixed integer programming with a large number of 0–1 variables, the bi-level discrete network design problem has been recognized as one of the most difficult yet challenging problems in transport [1, 12]. A large number of scholars have investigated NDP over the past three decades. Some of them proposed the continuous network design problems to avoid the complexity of NDP [1, 13, 14, 15]. This clearly simplified the problem because it removed the combinatorial aspects and made the problem amenable to a number of non-linear programming algorithms. The computational benefits of using continuous approximation to the discrete problem can be substantial [16]. The disadvantage to this approach is that it may be difficult to translate the continuous solution into particular projects which can be implemented. This is especially difficult if the continuous solution indicates relatively

small improvements on a large number of links. Because projects often incurred substantial start-up costs, such a solution may not be really practical [17].

This paper intends to develop a new DNDP, more consistent with the actual, where the variables can be a series of integers rather than just 0-1. Because the value range of variables is wider, the solution for the new DNDP is more complex. Leblanc in [18] used the branch-and-bound algorithm to solve DNDP, but he assumed that additional link improvements would always reduce the total user cost, the bounds were relatively "loose" and the value of variables can only be 0 or 1. Poorzahedy and Turnquist (1982) in [19] transformed the bi-level model into a single one through an approximation and then used the branch-and-bound algorithm to solve it, but the solution of the model may not be accurate. The methods proposed by Gao et al. in [17] and Hamid and Mohammad in [20], mainly dealt with 0-1 DNDP and could not be directly applied in solving the new DNDP with discrete values. Wang and Lo in [21] extended their algorithm to CNDP to solve the network design problems with discrete levels of capacity improvements, but the method could not deal with the case of new link additions and their basic idea was to transform CNDP into a mixed-integer linear program. Some algorithms for the integer program such as the branch-and-bound are still needed, so the solution is still very complex if the number of variables is large. Luathep et al. in [22] generalized the approach developed in Wang and Lo in [21] to solve any type of network design problems, including CNDP, DNDP and MNDP. The required constraints involve all the extreme points of the closed convex polyhedron for the feasible flow patterns. It is generally difficult to identify all the extreme points for a polyhedron, and in particular, for a moderately large network problem, the constraint set might become huge and intractable.

This paper discusses a new DNDP with discrete values. The solution will be very complex and time-consuming in the branch-and-bound method applied. In this paper each sub-problem of the branch is solved by the Hooke-Jeeves algorithm. The integer solutions are recorded in the searching process which lends itself to determine the upper bound on the upper-level objective function and cuts useless branches quickly, improving the speed of the branch-and-bound algorithm for earlier convergence. If the difference between the current lower bound and the upper bound is within a very small value, then the calculation ends. This rule can also be an additional rule for branch cutting, if the difference between the objective function value and the current upper bound is within a very small value. Then the sub-problem is cut.

The remainder of the paper is organized as follows. Section 2 presents the formulation of new DNDP. Section 3 proposes the solution algorithm combining branch-and-bound with Hooke-Jeeves algorithm. In Section 4 numerical examples are given to demonstrate the efficiency of the proposed method for this new DNDP. The final section concludes the paper and discusses the future research issues.

2. PROBLEM FORMULATION

2.1 Notations

The notations used throughout the paper are listed as follows, unless otherwise specified.

$\mathbf{G} = (\mathbf{N}, \mathbf{A})$	transportation network with $\mathbf{N}$ being the set of nodes and $\mathbf{A}$ ($\mathbf{A} = \mathbf{A_1} \cup \mathbf{A_2} \cup \mathbf{A_3}$) being the set of links, respectively
$\mathbf{R}$	set of origin nodes, where $\mathbf{R} \subset \mathbf{N}$
$\mathbf{S}$	set of destination nodes, where $\mathbf{S} \subset \mathbf{N}$
r	origin node index, where $r \in \mathbf{R}$
s	destination node index, where $s \in \mathbf{S}$
$\mathbf{A_1}$	set of non-expanded links, where $\mathbf{A_1} \subset \mathbf{A}$
$\mathbf{A_2}$	set of expanded links, where $\mathbf{A_2} \subset \mathbf{A}$
$\mathbf{A_3}$	set of new candidate links, where $\mathbf{A_3} \subset \mathbf{A}$
x_a	aggregate flow on link $a \in \mathbf{A}$
$\mathbf{x}$	vector whose elements are x_a
C_a^0	original capacity on existing link $a \in \mathbf{A_1} \cup \mathbf{A_2}$
y_a	discrete level of capacity improvements on expanded link $a \in \mathbf{A_2}$
$\mathbf{y}$	vector, whose elements are y_a
t_a	travel time of link $a \in \mathbf{A}$
$g_a(y_a)$	improvement cost function of expanded link $a \in \mathbf{A_2}$
u_a	discrete level of capacity on candidate link $a \in \mathbf{A_3}$, especially $u_a = 0$ if $a \in \mathbf{A_3}$ is not added into the network
$\mathbf{u}$	vector whose elements are u_a
ϕ	relative weight of travel time and construction expense
$d_a(u_a)$	construction cost function of new candidate link $a \in \mathbf{A_3}$
q_{rs}	travel demand between pair (r,s)
$\mathbf{q}$	OD (origin-destination) matrix, whose elements are q_{rs}
f_k^{rs}	flow of path k between pair (r,s)
$\mathbf{L}_{rs}$	set of paths between pair (r,s)
$\delta_{a,k}^{rs}$	path/link incidence variable, which equals 1 if link a is on path k between pair (r,s), otherwise 0

2.2 The upper-level optimization problem

In the actual road network design, it has to be determined which roads should be newly constructed and which roads need improvements. For the candi-

date new roads, it needs to be determined whether they should be added and what grade their capacities are if they are added. For the reconstructed roads, what grade should be decided to improve their capacities. There are some discrete levels rather than continuous values. Dividing the road capacity into a series of discrete grades is consistent, because for each link it may be one lane, or 2 lanes, 3 lanes, 4 lanes, etc., continuous variables cannot correspond well to the actual ones. Because the value range of variables is wider than just 0-1, the solution for this kind of DNDP with discrete values is more complex. The challenging problem is to find the ideal solution soon.

The new DNDP aims to find both capacity expansions of existing links (a series of discrete levels of capacity enhancements) and new link additions (not only to decide whether a new link is added, but also decide the grade of its capacity if it is added) in order to minimize the total travel time of the network users or to minimize the total cost including travel time and investment under the UE (user equilibrium) condition.

DNDPs have two types: DNDP having budget constraint and DNDP without budget constraint [18]. For DNDP having budget constraint, the upper-level optimization problem of DNDP is formulated as (1)~(6), where F is the total travel time of the network users.

$$\min_{\mathbf{y},\mathbf{u}} F(\mathbf{y},\mathbf{u},\mathbf{x}) = \sum_{a\in\mathbf{A}_1} x_a t_a(x_a) + \sum_{a\in\mathbf{A}_2} x_a t_a(x_a,y_a) + \\ + \sum_{a\in\mathbf{A}_3} x_a t_a(x_a,u_a) \tag{1}$$

subject to,

$$\sum_{a\in\mathbf{A}_2} g_a(y_a) + \sum_{a\in\mathbf{A}_3} d_a(u_a) \le budget \tag{2}$$

$$C_a^0 + y_a\theta_a \le \bar{C}_a, \quad \forall a\in\mathbf{A_2} \tag{3}$$

$$u_a\theta_a \le \bar{C}_a, \quad \forall a\in\mathbf{A}_3 \tag{4}$$

$$y_a \in \{0,1,2,3,\ldots\}, \quad \forall a\in\mathbf{A_2} \tag{5}$$

$$u_a \in \{0,1,2,3,\ldots\}, \quad \forall a\in\mathbf{A}_3 \tag{6}$$

where **x** is implicit function of **y** and **u** and can be obtained by solving the lower-level UE problem, θ_a is the capacity enhancement corresponding to each grade increase for link a and $\bar{C}_a$ is the upper bound of the capacity for link a. The budgetary constraint is presented in (2), the total expense including new link constructions and existing link reconstructions should be restricted within the budget. Constraint (3) expresses the upper bounds of the additional capacities of the expanded links. If $u_a = 0$ then $a(a\in\mathbf{A}_3)$ will not be added into the network. If $u_a = 1,2,3,\ldots$ then $a(a\in\mathbf{A_3})$ is added into the network and $u_a\theta_a$ is the capacity of new link a. Constraint (4) expresses the upper bounds of the capacities of new links. Constraints (5)~(6) show that $y_a(a\in\mathbf{A_2})$ and $u_a(a\in\mathbf{A}_3)$ are discrete variables whose values take 0,1,2,3,....

For DNDP without budget constraint, where the expense is placed into the objective function, the upper-level optimization problem of DNDP is formulated as (7)~(11), where F is the total cost including travel time and investment and ϕ is the relative weight of travel time and construction expense.

$$\min_{\mathbf{y},\mathbf{u}} F(\mathbf{y},\mathbf{u},\mathbf{x}) = \sum_{a\in\mathbf{A}_1} x_a t_a(x_a) + \sum_{a\in\mathbf{A}_2} x_a t_a(x_a,y_a) + \\ + \sum_{a\in\mathbf{A}_3} x_a t_a(x_a,u_a) + \phi\left[\sum_{a\in\mathbf{A}_2} g_a(y_a) + \sum_{a\in\mathbf{A}_3} d_a(u_a)\right] \tag{7}$$

subject to,

$$C_a^0 + y_a\theta_a \le \bar{C}_a, \quad \forall a\in\mathbf{A_2} \tag{8}$$

$$u_a\theta_a \le \bar{C}_a, \quad \forall a\in\mathbf{A}_3 \tag{9}$$

$$y_a \in \{0,1,2,3,\ldots\}, \quad \forall a\in\mathbf{A_2} \tag{10}$$

$$u_a \in \{0,1,2,3,\ldots\}, \quad \forall a\in\mathbf{A}_3 \tag{11}$$

where (8)~(11) are the same as (3)~(6).

Value θ_a emphasizes that road capacity is not proportional to adding of lanes. For example, the capacity of one lane is 500, while the capacity of 2 lanes is less than 1,000, about 900. The capacity of 3 lanes is less than 1,500, supposedly 1,200. So θ_a is not fixed for the above example: θ_a = 400 when the number of lanes increases from 1 to 2, while θ_a = 300 when the number of lanes increases from 2 to 3. This is mainly because the gap between vehicles in different lanes increases when lanes are addied But in this paper it will be assumed that θ_a is fixed for the convenience of the following modelling and calculation and θ_a will be the same whether the number of lanes increases from 1 to 2 or from 2 to 3.

2.3 The lower-level user equilibrium assignment

The UE problem with fixed demand can be formulated as (12)~(16) [23].

$$\min T(\mathbf{y},\mathbf{u},\mathbf{x}) = \sum_{a\in\mathbf{A}_1}\int_0^{x_a} t_a(w)dw + \sum_{a\in\mathbf{A}_2}\int_0^{x_a} t_a(w,y_a)dw + \\ + \sum_{a\in\mathbf{A}_3}\int_0^{x_a} t_a(w,u_a)dw \tag{12}$$

subject to,

$$\sum_{k\in\mathbf{L}_{rs}} f_k^{rs} = q_{rs}, \quad \forall r\in\mathbf{R}, s\in\mathbf{S} \tag{13}$$

$$x_a = \sum_r\sum_s\sum_k f_k^{rs}\cdot\delta_{a,k}^{rs}, \ \forall a\in\mathbf{A} \tag{14}$$

$$f_k^{rs} \ge 0, \quad \forall r\in\mathbf{R}, s\in\mathbf{S}, k\in\mathbf{L}_{rs} \tag{15}$$

$$x_a \le Mu_a, \quad a\in\mathbf{A}_3 \tag{16}$$

In this model, the users at the lower-level are assumed to follow the user-equilibrium principle of Wardrop under the given network. Constraints (13)~(15) are definitional and conservation of the flow constraints. Constraint (16) prohibits the flow on any proposed link that is not actually constructed and M is an

arbitrarily positive constant. If $u_a = 0$, then $x_a = 0$. If $u_a = 1, 2, 3, \ldots$ then x_a can be as large as desired.

3. THE SOLUTION ALGORITHM FOR THE BI-LEVEL PROBLEM

Because the value range of variables is wider (rather than just 0-1), the solution for DNDP with discrete values will be more complex and time-consuming. The previous studies about DNDP used variables which only took 0 or 1 [17, 18, 19, 20]. Wang and Lo in [21] extended their algorithm for CNDP to solve the network design problems with discrete levels of capacity improvements, but their method could not deal with the case of new link additions. Their basic idea was to transform CNDP into a mixed-integer linear program. Algorithms for the integer program such as the branch-and-bound were still needed, so the solution would be very complex if the number of variables were large.

The problems defined by (1)~(16) are essentially non-linear integer bi-level programs. This paper aims to develop an algorithm combining branch-and-bound with Hooke-Jeeves to solve it.

3.1 Proposed method idea

Hooke-Jeeves algorithm is a direct search method. Abdulaal and Leblanc in [16] applied the Hooke-Jeeves algorithm to solve CNDP. This method was a step search algorithm. The step length took a bigger initial value (an integer larger than or equal to 1) to search and then reduced by a certain ratio when it could not find a better solution and then continued to search until the step length was smaller than the given value. Then the solution could be obtained as the optimal one.

Branch-and-bound algorithm is a normal method for solving the integer or mixed integer program. Leblanc in [18] and Poorzahedy and Turnquist in [19] applied the branch-and-bound algorithm to solve 0-1 DNDP. This paper will design a solution method combining branch-and-bound with Hooke-Jeeves algorithm to solve the new DNDP with discrete values. Since the needed time would be large and calculation would be slow if only the branch-and-bound algorithm was used to solve the integer program with many variables, an attempt was made to find a better integer solution soon and thus save time. More branches can be cut whose objective function value after relaxation is larger than the one depending on this integer solution if a good integer solution has been obtained. The Hooke-Jeeves algorithm has been used to solve the relaxed problem (continuous relaxation). The Hooke-Jeeves algorithm has its advantage because if any initial feasible integer solution is given, then these solutions are all integer if search step length is an integer $\delta \geq 1$, and these feasible solutions will be better than the initial solution. When $\delta < 1$ the following solutions are more of non-integer solutions, but it may appear that all elements of a solution are integers at a step length smaller than 1. We must record each integer solution which may occur at each search step of Hooke-Jeeves algorithm in solving the relaxed problem and always replace the former ones with the latter better integer solutions. Always record the best integer solution as the new upper bound so far as any branch, whose objective function value after relaxation is larger than this upper bound, is cut.

In addition, for earlier convergence and saving the time, the ending rule should be set. Denote by F^* the optimal objective function value of the integer program (1)~(16) and suppose that the lower bound series of relaxed problems is produced as follows:

$$\underline{F}_1 \leq \underline{F}_2 \leq \ldots \leq \underline{F}_k \leq \ldots \leq F^*.$$

The relaxed problem removes (5), (6), (10), (11), i.e. the solutions of the relaxed problem may be non-integers. Low bounds $(\underline{F}_1, \underline{F}_2, \ldots, \underline{F}_k, \ldots)$ are a series of minimal objective function values of relaxed problems (the relaxed problems can add constraints (17)-(18) in the process of branch-and-bound algorithm, see next Section 3.2), so the values $(\underline{F}_1, \underline{F}_2, \ldots, \underline{F}_k, \ldots)$ are smaller than the objective function value (F^*) of the original integer problem (1)~(16) where the solutions can only be integers.

Upper bound series are

$$\overline{F}_1 \geq \overline{F}_2 \geq \ldots \geq \overline{F}_k \geq \ldots \geq F^*,$$

where $\overline{F}_1, \overline{F}_2, \ldots, \overline{F}_k$ are a series of objective function values of the integer program (1)~(16) corresponding to a series of integer solutions.

If $\overline{F}_k - \underline{F}_k \leq \varepsilon$ ($\varepsilon > 0$ and ε is a very small value), then

$$F^* \leq \overline{F}_k \leq F^* + \varepsilon,$$

for any feasible solution $\mathbf{y}^{(k)}$ of the integer program, $F(\mathbf{y}^{(k)}) = \overline{F}_k$. If $\overline{F}_k - \underline{F}_k \leq \varepsilon$, then the integer solution $\mathbf{y}^{(k)}$ is an ε approximately optimal solution.

The above rule can be applied in branch cutting. Suppose the current upper bound $\overline{F}$. Consider a branch whose objective function value (F) after relaxation satisfies $0 < \overline{F} - F \leq \varepsilon$ ($\varepsilon > 0$ and ε is a very small value). This branch has to be cut, without further rebranching (continuing to branch) of this branch. Because if the branch rebranches, its objective function value (F') of the integer solution will satisfy $F' \geq F$. Inequalities $\overline{F} - F' \leq \overline{F} - F \leq \varepsilon$ mean that even if a better feasible integer solution exists (having a smaller objective function value) when a branch rebranches, the difference between its value and the value of current integer solution will not surpass ε. For a very small ε, cutting branches like this can attain the precision need. If $\overline{F} - F < 0$, obviously this branch should be cut because the objective function value (F') of the inte-

ger solution of this branch satisfies $F' \geq \underline{F}$, from which $F' \geq \underline{F}$ can be derived. If $\overline{F} - F' < 0$ (i.e. $F' > \overline{F}$), obviously the better integer solution cannot be obtained, so this branch should be cut.

So the conditions for branch cutting are the following.

1. The optimal solution of the sub-problem (i.e. the branch) is an integer solution;
2. The sub-problem has no feasible solutions;
3. The objective function value of the sub-problem is larger than or equals the current upper bound;
4. The objective function value (F) of the sub-problem satisfies $\overline{F} - F \leq \varepsilon$, where $\overline{F}$ is the current upper bound.

3.2 The steps of the proposed algorithm

The steps of the proposed algorithm for solving the problem defined by (1)~(16) are as follows.

Denote the integer program defined by (1)~(16) as H and denote its continuous relaxation as B.

(1) Solving problem B by the Hooke-Jeeves algorithm, the cases as follows may occur:

① If problem B has no feasible solutions, then problem H also has no feasible solutions (because B is a continuous relaxation problem of H and $H \subseteq B$). End (because if any feasible solution does not exist, then any feasible integer solution certainly does not exist either).

② If problem B has an optimal integer solution, then it is just the optimal solution of problem H. End (the integer solution of B is also the solution of H because B is a continuous relaxation problem of H).

③ Problem B has a non-integer optimal solution, whose objective function value is $\underline{F}$. In the process of calculation with the Hooke-Jeeves algorithm, record the feasible best integer solution (having the smallest objective function value) as the upper bound ($\overline{F}$) on the upper objective function.

If F^* denotes the optimal objective function value of problem H, then $\underline{F} \leq F^* \leq \overline{F}$. Then perform the following:

Choose a variable (such as y_j) whose value is a non-integer (b_j) from the optimal solution of problem B. If $\lfloor b_j \rfloor$ denotes the maximal integer smaller than b_j, construct two constraints as follows:

$$y_j \leq \lfloor b_j \rfloor \tag{17}$$

$$y_j \geq \lfloor b_j \rfloor + 1. \tag{18}$$

Add each of the two constraints into problem B to obtain the following sub-problems $B1$ and $B2$ as the two branches. The initial solutions for solving sub-problems $B1$ and $B2$ takes on the j^{th} position of solution vector $\lfloor b_j \rfloor$ and $\lfloor b_j \rfloor + 1$, respectively. The other positions can take the same values as the optimal solution of previous program B. Record the feasible best integer solution in the search process of Hooke-Jeeves algorithm and round the optimal solutions of the followed sub-problems. If the rounded solutions are also feasible and better than the previous integer solutions, then they can be the best integer solutions in the whole calculation process.

Find the smallest objective function value as the new lower bound $\underline{F}$ from all current branches. From the branches whose optimal solutions are integer solutions and the best integer solutions in the search process of Hooke-Jeeves algorithm for other branches, find one which has the smallest objective function value and compare its objective function value with the last upper bound. If the value is smaller than the last upper bound then it can be the new upper bound. If it is larger than or equals to the last upper bound or no feasible integer solutions are obtained then the last upper bound remains the current upper bound.

Any branch whose optimal objective function value (F) is larger than or equals the upper bound $\overline{F}$ ($\overline{F} - F \leq 0$) or has no feasible solutions or satisfies $\overline{F} - F \leq \varepsilon$ ($\varepsilon > 0$ and ε is a very small value) is cut. For the branches whose optimal objective function value is smaller than $\overline{F}$ and the optimal solution is not an integer solution, tag the relaxed problem as B and repeat steps 1 and 2 until no branch on the tree needs to be considered or until the ending rule $\overline{F}_k - \underline{F}_k \leq \varepsilon$ is satisfied. The obtained best integer solution is just the optimal solution of the original problem H.

After branch cutting, the decision of choosing the branch to continue branching from the left branches in searching the tree also affects the convergence speed in the process of branching and bounding. Here the lower bound priority strategy is applied which means that always the branch with the smallest lower bound should be chosen to continue branching.

4. EXAMPLES

Example 1: The 16-link network.

The 16-link network, as shown in *Figure 1*, consists of six nodes and two O-D pairs. The arc numbers are on the links. All input information for these test networks are the same as those used by Suwansirikul et al. in [10]. The travel demands for the O-D pairs (1, 6) and (6, 1) are assumed to be q and $2q$, respectively. Travel demand levels with $q = 5$ and $q = 10$ are considered for the tests. The objective function of DNDP without budget constraint is

$$F = \sum_{a \in \mathbf{A}_2} x_a t_a(x_a, y_a) + \sum_{a \in \mathbf{A}_2} \eta_a y_a,$$

where η_a is improvement cost per unit incremental capacity of expanded link a ($a \in \mathbf{A}_2$), the relative weight (ϕ) of travel time and construction expense takes 1. Suppose that y_a, $a \in \mathbf{A}_2$ are integers between 0 and 6.

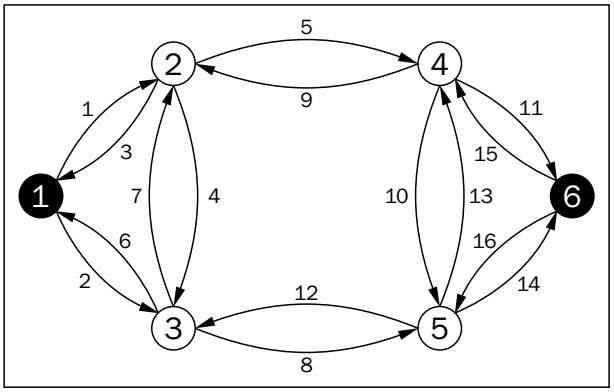

Figure 1 - The 16-link network

Table 1 - Link parameters for the 16-link network

$t_a(x_a,y_a) = \alpha_a + \beta_a(x_a/(C_a^0 + y_a))^4$, $a \in \mathbf{A}_2$								
link	node		α_a	β_a	C_a^0	η_a	**y**	**x**
	i	*j*						
1	1	2	1	10	3	2	y_1	x_1
2	1	3	2	5	10	3	y_2	x_2
3	2	1	3	3	9	5	y_3	x_3
4	2	3	4	20	4	4	y_4	x_4
5	2	4	5	50	3	9	y_5	x_5
6	3	1	2	20	2	1	y_6	x_6
7	3	2	1	10	1	4	y_7	x_7
8	3	5	1	1	10	3	y_8	x_8
9	4	2	2	8	45	2	y_9	x_9
10	4	5	3	3	3	5	y_{10}	x_{10}
11	4	6	9	2	2	6	y_{11}	x_{11}
12	5	3	4	10	6	8	y_{12}	x_{12}
13	5	4	4	25	44	5	y_{13}	x_{13}
14	5	6	2	33	20	3	y_{14}	x_{14}
15	6	4	5	5	1	6	y_{15}	x_{15}
16	6	5	6	1	4.5	1	y_{16}	x_{16}

Figure 2 shows the branch-and-bound tree when $q = 5$ and $\varepsilon = 0.001$. The numbers beside the ovals are branch numbers. F1 denotes the optimal objective function value of the relaxed problem (The relaxed problem removes (10) and so its solutions may be non-integers, and the relaxed problem can add constraints (17)-(18) in the process of branch-and-bound algorithm), F denotes the function value of the obtained best integer solution in the search process of Hooke-Jeeves algorithm. The Hooke-Jeeves algorithm is applied to solve each sub-problem with initial search step size $\delta = 1$ and decline factor 0.5. Branches 3, 4 and 5 have had integer solutions, so the integer solution of the original problem is **y** = [0 0 0 0 0 5 0 0 0 0 0 0 0 0 0 6], the objective function value (F) is 200.3915, and the corresponding x can be obtained from the UE equilibrium assignment ((12)~(16)), **x** = [0, 5, 6.1074, 0, 0, 3.8926, 0, 5, 6.1074, 0, 0, 3.8926, 5.0843, 5, 1.0231, 8.9769]. It is noticeable that the optimal integer solution has been found on Branch 1.

Figure 3 shows the branch-and-bound tree when $q = 10$. $\varepsilon = 0.001$. The best integer solution is **y** = [0 5 6 0 0 6 0 1 0 0 0 0 0 1 6 6], the objective function value (F) is 588.2846, and the corresponding x can be obtained from the UE equilibrium assignment ((12)~(16)), **x** = [0, 10, 15.2513, 0, 0, 4.7487, 0, 10, 15.2513, 0, 0, 4.7487, 7.6599, 10, 7.5914, 12.4086]. It is noticeable that the optimal integer solution has been found on Branch 1. If $\varepsilon = 0.01$ as threshold for branch cutting and ending, then just a step on Branch 1 is enough to find the best integer solution.

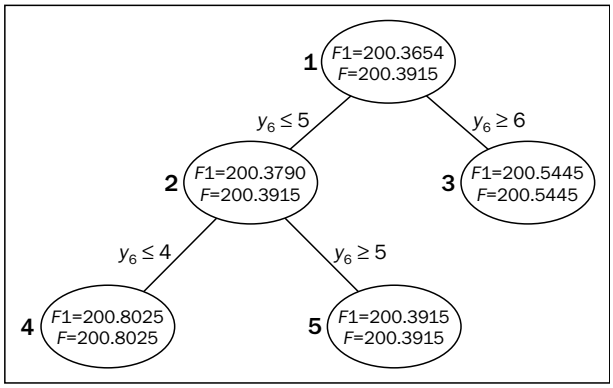

Figure 2 - Branch-and-bound tree when q = 5

Table 2 - Objective function values and solutions in the branching when q = 5

Branch	Objective function value		Solution (y)
1	*F*1	200.3654	[0, 0, 0, 0, 0, 5.25, 0, 0, 0, 0, 0, 0, 0, 0, 0, 6]
	F	200.3915	[0, 0, 0, 0, 0, 5, 0, 0, 0, 0, 0, 0, 0, 0, 0, 6]
2	*F*1	200.3790	[0, 0, 0, 0, 0, 4.75, 0, 0, 0, 0, 0, 0, 0, 0, 0, 6]
	F	200.3915	[0, 0, 0, 0, 0, 5, 0, 0, 0, 0, 0, 0, 0, 0, 0, 6]
3	*F*1	200.5445	[0, 0, 0, 0, 0, 6, 0, 0, 0, 0, 0, 0, 0, 0, 0, 6]
	F	200.5445	[0, 0, 0, 0, 0, 6, 0, 0, 0, 0, 0, 0, 0, 0, 0, 6]
4	*F*1	200.8025	[0, 0, 0, 0, 0, 4, 0, 0, 0, 0, 0, 0, 0, 0, 0, 6]
	F	200.8025	[0, 0, 0, 0, 0, 4, 0, 0, 0, 0, 0, 0, 0, 0, 0, 6]
5	*F*1	200.3915	[0, 0, 0, 0, 0, 5, 0, 0, 0, 0, 0, 0, 0, 0, 0, 6]
	F	200.3915	[0, 0, 0, 0, 0, 5, 0, 0, 0, 0, 0, 0, 0, 0, 0, 6]

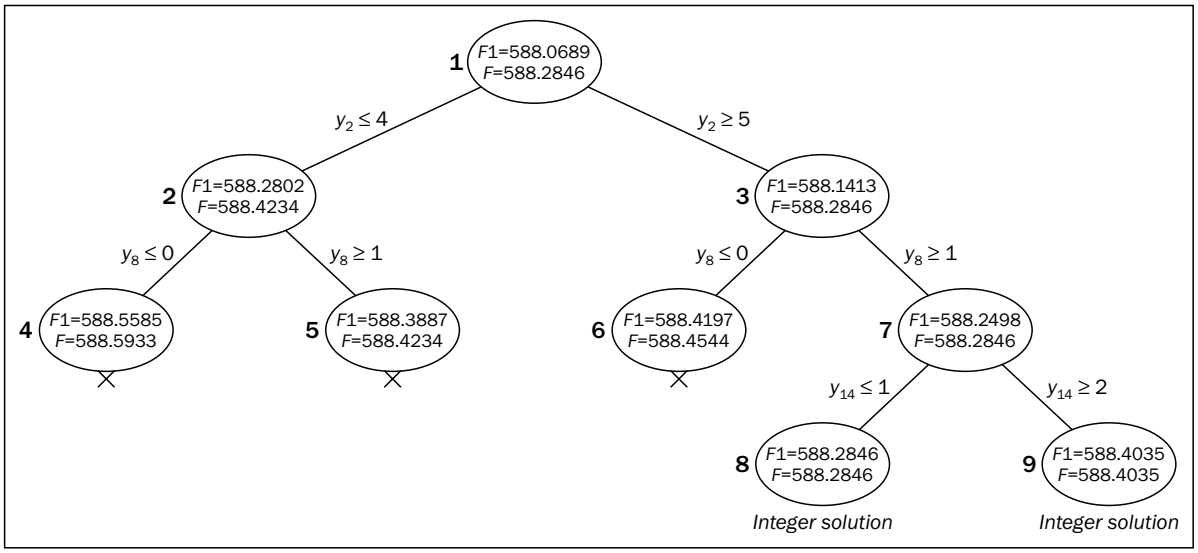

Figure 3 - Branch-and-bound tree when q = 10

Table 3 - Objective function values and solutions in the branching when q = 10

Branch	Objective function value		Solution (y)
1	*F*1	588.0689	[0, 4.625, 6, 0, 0, 6, 0, 0.625, 0, 0, 0, 0, 0, 1.375, 6, 6]
	F	588.2846	[0, 5, 6, 0, 0, 6, 0, 1, 0, 0, 0, 0, 0, 1, 6, 6]
2	*F*1	588.2802	[0, 4, 6, 0, 0, 6, 0, 0.625, 0, 0, 0, 0, 0, 1.375, 6, 6]
	F	588.4234	[0, 4, 6, 0, 0, 6, 0, 1, 0, 0, 0, 0, 0, 1, 6, 6]
3	*F*1	588.1413	[0, 5, 6, 0, 0, 6, 0, 0.625, 0, 0, 0, 0, 0, 1.375, 6, 6]
	F	588.2846	[0, 5, 6, 0, 0, 6, 0, 1, 0, 0, 0, 0, 0, 1, 6, 6]
4	*F*1	588.5585	[0, 4, 6, 0, 0, 6, 0, 0, 0, 0, 0, 0, 0, 1.375, 6, 6]
	F	588.5933	[0, 4, 6, 0, 0, 6, 0, 0, 0, 0, 0, 0, 0, 1, 6, 6]
5	*F*1	588.3887	[0, 4, 6, 0, 0, 6, 0, 1, 0, 0, 0, 0, 0, 1.375, 6, 6]
	F	588.4234	[0, 4, 6, 0, 0, 6, 0, 1, 0, 0, 0, 0, 0, 1, 6, 6]
6	*F*1	588.4197	[0, 5, 6, 0, 0, 6, 0, 0, 0, 0, 0, 0, 0, 1.375, 6, 6]
	F	588.4544	[0, 5, 6, 0, 0, 6, 0, 0, 0, 0, 0, 0, 0, 1, 6, 6]
7	*F*1	588.2498	[0, 5, 6, 0, 0, 6, 0, 1, 0, 0, 0, 0, 0, 1.375, 6, 6]
	F	588.2846	[0, 5, 6, 0, 0, 6, 0, 1, 0, 0, 0, 0, 0, 1, 6, 6]
8	*F*1	588.2846	[0, 5, 6, 0, 0, 6, 0, 1, 0, 0, 0, 0, 0, 1, 6, 6]
	F	588.2846	[0, 5, 6, 0, 0, 6, 0, 1, 0, 0, 0, 0, 0, 1, 6, 6]
9	*F*1	588.4035	[0, 5, 6, 0, 0, 6, 0, 1, 0, 0, 0, 0, 0, 2, 6, 6]
	F	588.4035	[0, 5, 6, 0, 0, 6, 0, 1, 0, 0, 0, 0, 0, 2, 6, 6]

Example 2: The Sioux Falls network

The second test is with the network of Sioux Falls city, as shown in *Figure 4*. The objective function of DNDP is

$$F = \sum_{a \in \mathbf{A}} x_a t_a(x_a) + 0.001 \sum_{a \in \mathbf{A}_2} \lambda_a y_a^2,$$

where $\lambda_a y_a^2$ is the improvement cost function of expanded link $a \in \mathbf{A}_2$, and the relative weight (ϕ) of travel time and construction expense takes 0.001. The link data and the O-D travel demands between 552 O-D pairs are the same as those used by Suwansirikul et al. in [10]. Here, the only difference is that it must be an integer solution between 0 and 6. Note that there is no budget constraint for DNDP. ε = 0.08 as threshold for branch cutting and ending. It is supposed that links needing to be expanded are shown in *Table 4*.

In *Figure 4*, Branch 3 is cut because the difference between the objective function value F (81.4551) and the current upper bound (81.5190) is 0.0639 which is smaller than ε (0.08). In the same way, Branches 5 and 7 are cut. Branch 6 is cut because the objective function value F (81.5892) is larger than the upper bound (81.5190). So the best integer solution of the original problem is **y** = [6 2 5 2 3 2 3 5 4 5], the objective function value is 81.5190. In fact, on Branch

Table 4 - Links needing to be expanded

node	*i*	6	7	8	8	9	10	10	13	16	24
	j	8	8	6	7	10	9	16	24	10	13
y		y_1	y_2	y_3	y_4	y_5	y_6	y_7	y_8	y_9	y_{10}

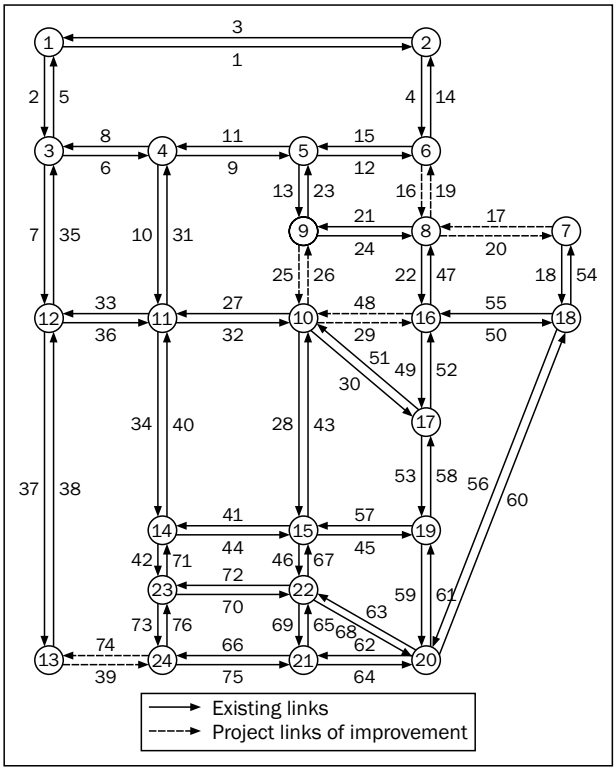

Figure 4 - The Sioux Falls network

1 the optimal integer solution has been found. It is noticeable that if ε takes a very small value, such as 0.000001, then the optimal solution is **y** = [5 2 5 2 2 3 3 5 4 5], the objective function value is 81.4737. Do not draw the detailed branch-and-bound tree due to the restriction of text length.

Example 3: The 12-node network

The 12-node network, as shown in *Figure 6*, was used by Gao et al. in [17]. It consists of twelve nodes and one OD pair (1, 12). The solid lines are the existing links whereas six candidate links are represented by the dashed lines. The travel time function on each link is assumed to be $t_a(x_a) = t_a^0 + 0.15(x_a/C_a)^4$, where t_a^0 is the free flow travel time on link a, C_a is the capacity on link a. For the existing links $a \in \mathbf{A}_1$, let $C_a = 2.0809$ ($a \in \mathbf{A}_1$), so $t_a(x_a) = t_a^0 + 0.008x_a^4$ ($a \in \mathbf{A}_1$). For six candidate links $a \in \mathbf{A}_3$, let $\theta_a = 2.0809$ (θ_a is the capacity enhancement corresponding to each grade increase for link a), u_a ($a \in \mathbf{A}_3$) is the variable and $u_a \in \{0,1,2,3,\ldots\}$, $C_a = u_a\theta_a$ ($a \in \mathbf{A}_3$). t_a^0 of the existing links are labeled beside the links, whereas t_a^0 of the candidate links are given as the first value in parentheses. The construction cost function is $d_a(u_a) = u_a\sigma_a$, where σ_a is the needed construction cost of capacity enhancement θ_a, shown as the second value in parentheses for each candidate link. The total O-D demand is assumed to be 20.

(1) $u_a \in \{0,1,2,3\}$, budget =100, $\varepsilon = 10$ as threshold for branch cutting and ending. *Figure 7* presents the branch-and-bound tree.

For the solution of NDP with budget constraint, the penalty function method in [24] is first used to transform it (NDP with budget constraint) to an extremum problem without constraints.

Here, $\varepsilon = 10$ as threshold for branch cutting and ending and the accuracy is 10/1874.6 = 0.0053. Branch 3 is cut because the difference between the

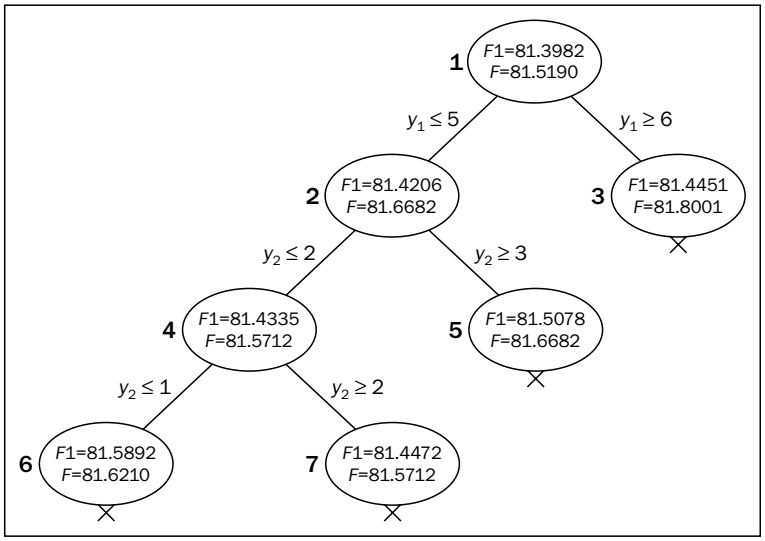

Figure 5 - Branch-and-bound tree of the Sioux Falls network

Table 5 - Objective function values and solutions in the branching of the Sioux Falls network

Branch	Objective function value		Solution (y)
1	*F*1	81.3982	[5.5, 2, 5.5, 2, 3, 2, 3.125, 5.125, 4, 5]
	F	81.5190	[6, 2, 5, 2, 3, 2, 3, 5, 4, 5]
2	*F*1	81.4206	[5, 2.5, 5.5, 2, 3, 2, 3.125, 5.125, 4, 5]
	F	81.6682	[5, 3, 6, 2, 3, 2, 3, 5, 4, 5]
3	*F*1	81.4451	[6, 2, 5.5, 2, 3.75, 2, 3.125, 4.625, 4, 5]
	F	81.8001	[6, 2, 6, 2, 4, 2, 3, 5, 4, 5]
4	*F*1	81.4335	[5, 1.75, 5, 2, 3, 2, 3.125, 5.125, 4, 5]
	F	81.5712	[5, 2, 5, 2, 3, 2, 3, 5, 4, 5]
5	*F*1	81.5078	[5, 3, 5.5, 2, 3, 2, 3.125, 5.125, 4, 5]
	F	81.6682	[5, 3, 6, 2, 3, 2, 3, 5, 4, 5]
6	*F*1	81.5892	[4.75, 1, 6, 1.75, 3, 2, 3.125, 5.125, 4, 5]
	F	81.6210	[5, 1, 6, 2, 3, 2, 3, 5, 4, 5]
7	*F*1	81.4472	[5, 2, 5, 2, 3, 2, 3.25, 5.125, 4, 5]
	F	81.5712	[5, 2, 5, 2, 3, 2, 3, 5, 4, 5]

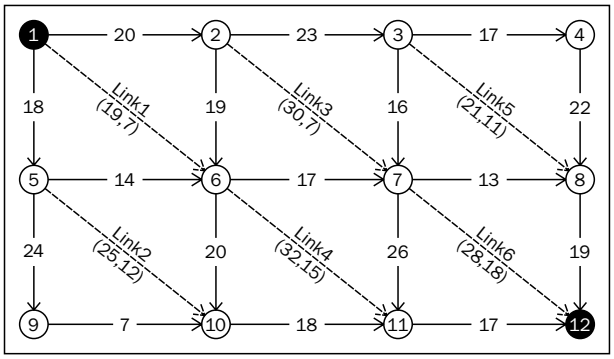

Figure 6 - The 12-node network

objective function values (1870.8) and the current upper bound (1874.6) is 3.8 which is smaller than ε (10). In the same way, branches 5 and 8 are cut. Branch 7 is cut because the objective function value (1878.6) is larger than the upper bound (1874.6), and in the same way branches 10 and 11 are cut. So the objective function value is 1874.6 and the corresponding optimal integer solution is **u** = [3 1 2 1 0 2].

(2) $u_a = \{0, 1\}$ budget =50.

The results in [17] are obtained under $u_a = \{0, 1\}$ so the research in [17] is a special case of the research above. Let ε = 10 as threshold for branch cutting and ending. *Figure 8* presents the simplified branch-and-bound tree. On branch 2 an integer solution was obtained and its function value was 2449.3. On branch 5 a better integer solution was obtained and the function value was 2406.2 as new upper bound. Branches 2 and 7 are cut because the objective function values (2446.5, 2633.0) are larger than the upper bound (2406.2), while branches 4 and 6 were cut because the differences between their objective function values (2401.5, 2402.5) and the current upper bound (2406.2) are smaller than ε (10). So the optimal solution of the original problem is [1 0 1 1 0 1], the objective function value is 2406.2, which is the same as the one reported in [17].

5. CONCLUSION AND DISCUSSION

A new kind of DNDP was proposed in this paper on the basis of the existing 0-1 DNDP, where the variables can be a series of integers rather than just 0-1. The new DNDP is consistent with the practical projects because whether it is a newly constructed road or a reconstructed one, its capacity needs to be determined in the first place. Because the capacity of the road can only be some discrete levels corresponding to the number of lanes it has in the practical projects, it is more realistic to use DNDP with discrete values than CNDP.

Since the value range of variables is wider than just 0-1, the solution for DNDP with discrete values is more complex. This paper designed a solution method combining branch-and-bound with Hooke-Jeeves algorithm. Branch-and-bound algorithm is a normal method for solving the integer or mixed integer program. Hooke-Jeeves algorithm is a step searching algorithm. The step length takes a bigger initial value to search and then reduces by a certain ratio until a solution satisfying accuracy is obtained. Through a combination of branch-and-bound and Hooke-Jeeves algorithm, the relaxed problem of each sub-problem can be solved by the Hooke-Jeeves algorithm. Feasible integer solutions are recorded in the search process which lends itself to determine the upper bound on the upper-level objective function and cut useless branches quickly, improving the speed of the branch-and-bound algorithm. For earlier convergence, if the difference between the current lower bound and the upper bound is within a very small value, then the calculation ends and the current upper bound is the approximate optimal ob-

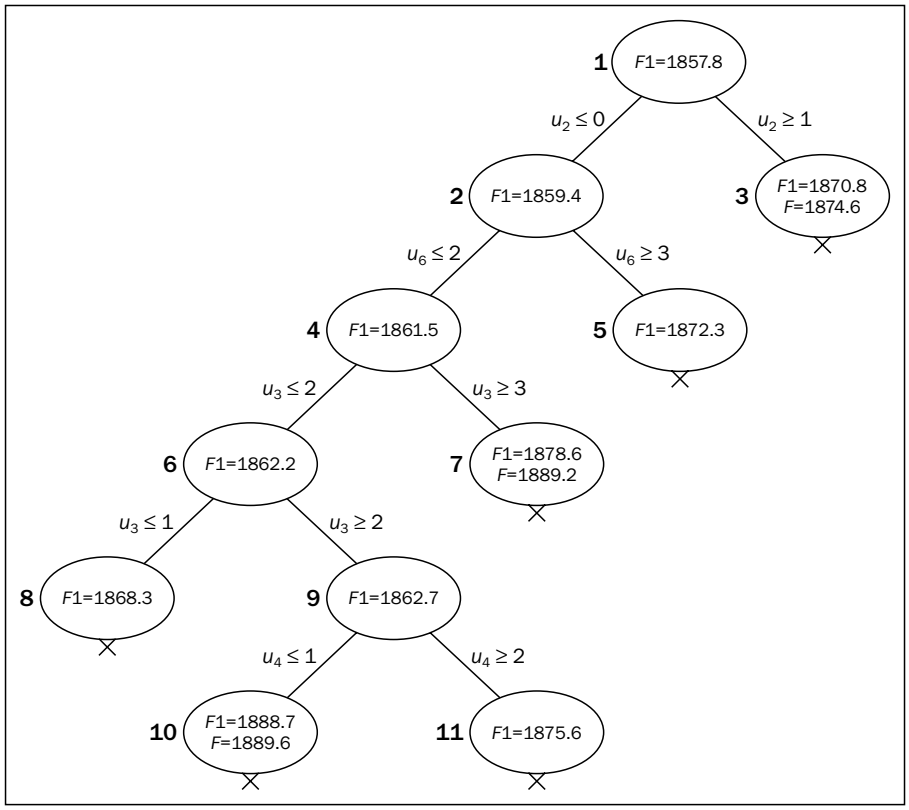

Figure 7 - Branch-and-bound tree when $u_a \in \{0, 1, 2, 3\}$

Table 6 - Objective function values and solutions in the branching when $u_a = \{0,1,2,3\}$

Branch	Objective function value		Solution (u)
1	*F*1	1857.8	[3, 0.5, 2.25, 0.625, 0.25, 2.5]
	F	-	no feasible integer solution
2	*F*1	1859.4	[3, 0, 2.75, 0.875, 0, 2.5]
	F	-	no feasible integer solution
3	*F*1	1870.8	[3, 1.125, 1.625, 0.75, 0.125, 2.25]
	F	1874.6	[3, 1, 2, 1, 0, 2]
4	*F*1	1861.5	[3, 0, 2.125, 1.875, 0, 2]
	F	-	no feasible integer solution
5	*F*1	1872.3	[2.875, 0, 1.75, 0.875, 0, 3]
	F	-	no feasible integer solution
6	*F*1	1862.2	[3, 0, 1.875, 1.875, 0.125, 2]
	F	-	no feasible integer solution
7	*F*1	1878.6	[2.875, 0, 3, 1.5, 0, 2]
	F	1889.2	[3, 0, 3, 1, 0, 2]
8	*F*1	1868.3	[3, 0, 1, 1.75, 0.875, 2]
	F	-	no feasible integer solution
9	*F*1	1862.7	[3, 0, 2, 1.875, 0, 2]
	F	-	no feasible integer solution
10	*F*1	1.8887	[3, 0, 2, 1, 0.125, 2]
	F	1.8896	[3, 0, 2, 1, 1, 2]
11	*F*1	1.8756	[2.75, 0, 2, 2, 0, 2]
	F	-	no feasible integer solution

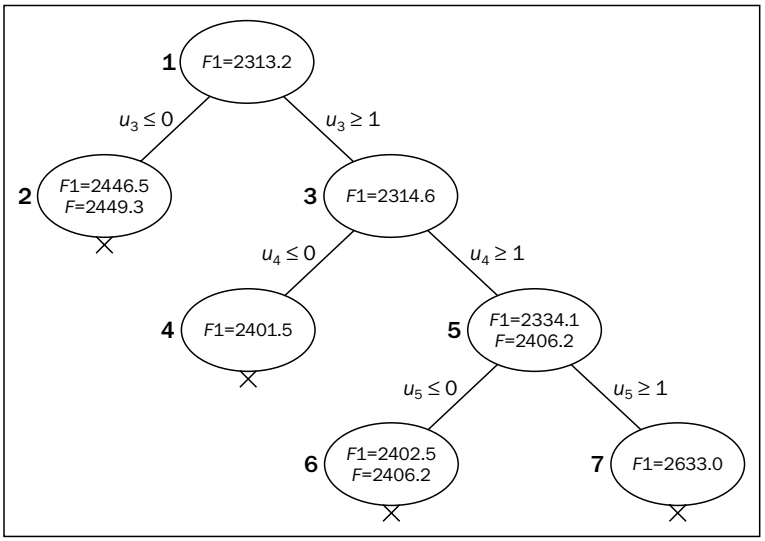

Figure 8 - Branch-and-bound tree when $u_a \in \{0, 1\}$

Table 7 - Objective function values and solutions in the branching when $u_a = \{0,1\}$

Branch	Objective function value		Solution (u)
1	*F*1	2313.2	[1, 0, 0.75, 0.75, 0.75, 1]
	F	-	no feasible integer solution
2	*F*1	2446.5	[1, 0, 0, 0.75, 1, 1]
	F	2449.3	[1, 0, 0, 1, 1, 1]
3	*F*1	2314.6	[1, 0.125, 1, 0.5, 0.75, 1]
	F	-	no feasible integer solution
4	*F*1	2401.5	[1, 0.625, 1, 0, 0.875, 1]
	F	-	no feasible integer solution
5	*F*1	2334.1	[1, 0, 1, 1, 0.375, 1]
	F	2406.2	[1, 0, 1, 1, 0, 1]
6	*F*1	2402.5	[1, 0.25, 1, 1, 0, 1]
	F	2406.2	[1, 0, 1, 1, 0, 1]
7	*F*1	2633.0	[0.75, 0, 1, 1, 1, 0.625]
	F	-	no feasible integer solution

jective function value. This rule can also be an additional rule for branch cutting. If the difference between the objective function value and the current upper bound is within a very small value, this sub-problem is cut. Four numerical examples are given to demonstrate the efficiency of the proposed method for this new DNDP.

For DNDP with discrete values, because the value range of variables is wide, it is very time-consuming to solve it using the branch-and-bound algorithm if the number of links is large. Although a solution method of combining branch-and-bound and Hooke-Jeeves algorithm was proposed in this paper with an addition of convergence and branch cutting rules for improving the calculation speed, only a local optimal solution can be obtained to solve the relaxed problems because the Hooke-Jeeves algorithm is only a kind of local optimal algorithm. Finding the algorithms which cannot only give the global optimal solution, but also record feasible integer solutions obtained in the search process and combine it with branch-and-bound algorithm to solve DNDP with discrete values is still an issue for future research.

ACKNOWLEDGEMENTS

The first author would like to thank the University of Leeds for supporting the visiting academic research. This research is also supported by the National Natural Science Foundation of China (Grant No. 50908235) and the Hunan Province Social Science Fund (Grant No. 12YBB274).

陈群，博士，副教授，单位：
中国中南大学交通运输工程学院
陈海波，博士，主任研究员，单位：
英国利兹大学交通研究中心

摘要

种新的双层离散网络设计问题的求解算法

提出一种新的离散网络设计模型，其变量值不再是0-1，而是可取多个水平。这种新的离散网络模型可同时确定改扩建路段的通行能力提高等级及新建路段选址与通行能力等级，在满足预算约束和用户均衡条件下使得网络中总的行驶时间最小。设计了一种分支定界和Hooke-Jeeves相结合的算法，在每个子问题用Hooke-Jeeves算法对其松弛规划进行求解的时候，记录其步长搜索过程中得到的整数解，这样可较快确定问题上界。为了尽快收敛并定义了剪枝和结束的准则。通过算例验证了该算法的有效性。

关键词

离散网络设计问题；整数规划；分支定界算法

REFERENCES

[1] **Yang, H., Bell, M.G.H.**: *Models and algorithms for road network design: a review and some new developments*, Transport Reviews, Vol. 18, No. 3, 1998, pp. 257-278

[2] **Chen, M.Y., Alfa, A.S.**: *A network design algorithm using a stochastic incremental traffic assignment approach*, Transportation Science, Vol. 25, No. 3, 1991, pp. 214-224

[3] **Davis, G.A.**: *Exact local solution of the continuous network design problem via stochastic user equilibrium assignment*, Transportation Research Part B, Vol. 28, No. 1, 1994, pp. 61–75

[4] **Patriksson, M.**: *On the applicability and solution of bi-level optimization models in transportation science: a study on the existence, stability and computation of optimal solutions to stochastic mathematical programs with equilibrium constraints*. Transportation Research Part B, Vol. 42, No. 10, 2008, pp. 843–860

[5] **Chiou, S.W.**: *Bilevel programming for the continuous transport network design problem*, Transportation Research Part B, Vol. 39, No.4, 2005, pp. 361–383

[6] **Meng, Q., Yang, H., Bell, M.G.H.**: *An equivalent continuously differentiable model and a locally convergent algorithm for the continuous network design problem*, Transportation Research Part B, Vol. 35, No. 1, 2001, pp. 83–105

[7] **Sumalee, A., Watling, D.P., Nakayama, S.**: *Reliable network design problem: case with uncertain demand and total travel time reliability*, Transportation Research Record, Vol. 1964, 2006, pp. 81–90

[8] **Wong, S.C., Yang, H.**: *Reserve capacity of a signal-controlled road network*, Transportation Research Part B, Vol. 31, No.5, 1997, pp. 397–402

[9] **Friesz, T.L., Harker, P.T.**: *Properties of the iterative optimization-equilibrium algorithm*, Civil Engineering Systems, Vol. 2, No.3, 1985, pp. 142-154

[10] **Suwansirikul, C., Friesz, T.L., Tobin, R.L.**: *Equilibrium decomposed optimization: a heuristic for the continuous equilibrium network design problems*, Transportation Science, Vol. 21, No. 4, 1987, pp. 254–263

[11] **Marcotte, P., Marquis, G.**: *Efficient implementation of heuristic for the continuous network design problems*, Annals of Operation Research, Vol. 34, No.1, 1992, pp. 163-176.

[12] **Magnanti, T.L., Wong, R.T.**: *Network design and transportation planning: models and algorithms*, Transportation Science, Vol. 18, No.1, 1984, pp. 1-55

[13] **Dantzig, G.B., Maier, S.F., Lansdowne, Z.F.**: *Application of decomposition to transportation network analysis*, Control Analysis Corporation, Palo Alto, California, Technical Report No. 1, March 1976

[14] **Boyce, D.E.**: *Urban transportation network equilibrium and design models: Recent achievements and future prospectives*, Environment and Planning Part A, Vol. 16, No. 1, 1984, pp. 1445–1474

[15] **Connors, R.D., Sumalee, A., Watling, D.P.**: *Sensitivity analysis of the variable demand probit stochastic user equilibrium with multiple user-classes*, Transportation Research Part B, Vol. 41, No. 6, 2007, pp. 593–615

[16] **Abdulaal, M., Leblanc, L.J.**: *Continuous equilibrium network design models*, Transportation Research Part B, Vol. 13, No. 1, 1979, pp. 19–32

[17] **Gao, Z., Wu, J., Sun, H.**: *Solution algorithm for the bi-level discrete network design problem*, Transportation Research Part B, Vol. 39, No. 6, 2005, pp. 479–495

[18] **Leblanc, L.J.**: *An algorithm for the discrete network design problem*, Transportation Science, Vol. 9, No. 3, 1975, pp. 183–199

[19] **Poorzahedy, H., Turnquist, M.A.**: *Approximate algorithms for the discrete network design problem*, Transportation Research Part B, Vol. 16, No. 1, 1982, pp. 45–55

[20] **Hamid, F., Mohammad, M.S.**: *A single-level mixed integer linear formulation for a bi-level discrete network design problem*, Transportation Research Part E, Vol. 47, No. 5, 2011, pp. 623–640

[21] **Wang, D.Z.W., Lo, H.K.**: *Global optimum of the linearized network design problem with equilibrium flows*, Transportation Research Part B, Vol. 44, No.4, 2010, pp. 482–492

[22] **Luathep, P., Sumalee, A., Lam, W.H.K., Li, Z., Lo, H.K.**: *Global optimization method for mixed transportation network design problem: A mixed-integer linear programming approach*, Transportation Research Part B, Vol. 45, No. 5, 2011, pp. 808-827

[23] **Sheffi, Y.**: *Urban Transportation Networks: Equilibrium Analysis with Mathematical Programming Methods*, Prentice-Hall, Englewood Cliffs, NJ, USA, 1985

[24] **Mokhtar, S.B., Shetty, C.M.**: *Nonlinear Programming: Theory and Algorithms*, John Wiley & Sons, Inc., New York, 1979