J Comb Optim
DOI 10.1007/s10878-015-9962-y

Multi-neighborhood based iterated tabu search for routing and wavelength assignment problem

Xinyun Wu[1] · **Shengfeng Yan**[1] · **Xin Wan**[1] · **Zhipeng Lü**[1]

Abstract The routing and wavelength assignment problem (RWA) has shown to be NP-hard if the wavelength continuity constraint and the objective of minimizing the number of wavelengths are considered. This paper introduces a multi-neighborhood based iterated tabu search algorithm (MN-ITS), which consists of three neighborhoods with a unified incremental evaluation method, to solve the min-RWA problem. The proposed MN-ITS algorithm is tested on a set of widely studied real world instances as well as a set of challenging random ones in the literature. Comparison with other reference algorithms shows that the MN-ITS algorithm is able to improve five best upper bounds obtained by other competitive reference algorithms in the literature. This paper also presents an analysis to show the significance of the unified incremental evaluation technique and the combination of multiple neighborhoods.

Keywords Routing and wavelength assignment · Network design · Multi-neighborhood search · Tabu search · Perturbation operator

1 Introduction

The routing and wavelength assignment (RWA) problem is introduced in the wavelength division multiplexing (WDM) networks. In WDM networks data streams are carried between nodes through a lightpath, which can be recognized as a channel connecting two nodes by a sequence of fiber links. By assigning each lightpath a unique wavelength, multiple data streams can traverse the same optical fiber without mutual

✉ Zhipeng Lü
zhipeng.lv@hust.edu.cn

[1] SMART, School of Computer Science and Technology, Huazhong University of Science and Technology, Wuhan 430074, People's Republic of China

Published online: 27 October 2015

interference, which is the main superiority of the WDM networks comparing to the electrical networks. This characteristic makes the WDM network have the ability to provide users with very large bandwidths (Dutta and Rouskas 2000).

In real applications, there is a limit for the number of wavelengths on each fiber. Consequently, the wavelengths become limited resources and will become extremely expensive as the data stream expands rapidly. To solve this problem, the min-RWA problem has been widely studied in the literature.

The min-RWA problem deals with the routing of lightpaths and assignment of wavelengths to lightpaths between pairs of nodes. The wavelength continuity constraint represents that a single wavelength must be assigned to a given lightpath. Therefore, a particular wavelength cannot be assigned to two different lightpaths which share a common physical link in order to prevent the data stream from interfering each other.

In this paper, we propose a multi-neighborhood based iterated tabu search algorithm (MN-ITS) for solving the min-RWA problem. The proposed MN-ITS algorithm integrates several distinguishing features, such as a multi-neighborhood search strategy, a unified fast incremental evaluation technique to evaluate multiple neighborhoods, a tabu search procedure and a perturbation operator to guide the search to jump out of local optimum trap. Our MN-ITS algorithm is tested on two sets of benchmark instances in the literature and shows its efficacy in terms of both solution quality and efficiency.

The paper is structured as follows. In Sect. 2 we present the mathematical formulation of the min-RWA problem and discuss related works in the literature. In Sect. 3 the proposed MN-ITS algorithm for solving the min-RWA problem is described in detail. Computational results and comparisons with other reference algorithms are presented in Sect. 4, and two important ingredients of the proposed algorithm are discussed and analyzed in Sect. 5. Finally, Sect. 6 concludes the paper.

2 Problem formulation and related works

2.1 Mathematical formulation of min-RWA

Given a physical network and a set of lightpath requests, the aim of the min-RWA problem is to find an assignment of each lightpath with a wavelength under the constraint that any two lightpaths with the same wavelength are routed through edge disjoint paths in the physical network, such that the number of used wavelengths is minimized. Note that there may be two requests with the same source-sink nodes. In this case, they should be routed through two edge disjoint paths or be assigned to different wavelengths.

Formally, the min-RWA can be formulated as an integer linear program (see Zang et al. (2000)) where the objective is to minimize the number of used wavelengths. Let $G = (V, E)$ denote the physical network where V is the set of nodes and E is the set of bidirectional arcs, and Λ the lightpath requests matrix where Λ_{sd} denotes the number of lightpaths between source node s and destination node d. Decision variables λ_{sdw} and F_{ij}^{sdw} respectively represent the number of lightpaths from source node s to destination node d assigned to wavelength w and the number of lightpaths from s to d with wavelength w on link (i, j). Then, the min-RWA problem can be mathematically stated as:

Minimize: F_{max}

Subject to:

$$\sum_{s,d,w} F_{ij}^{sdw} \leq F_{max} \quad \forall (i, j) \in E \tag{1}$$

$$\sum_{i} F_{ij}^{sdw} - \sum_{k} F_{jk}^{sdw} = \begin{cases} -\lambda_{sdw}, & \text{if } j = s; \\ \lambda_{sdw}, & \text{if } j = d; \\ 0, & \text{otherwise.} \end{cases} \quad \forall w, s, d, j \in V \tag{2}$$

$$\sum_{w} \lambda_{sdw} = \Lambda_{sd} \quad \forall s, d \tag{3}$$

$$\sum_{s,d} F_{ij}^{sdw} \leq 1 \quad \forall w, (i, j) \in E \tag{4}$$

$$F_{ij}^{sdw} = 0, 1 \quad \forall w, s, d, (i, j) \in E \tag{5}$$

Constraints (1) ensure that the objective value is no larger than the maximum number of lightpaths on one link. Constraints (2) guarantee that each lightpath has a feasible route. Constraints (3) require that the numbers of lightapths match the requests, and constraints (4) impose that the lightpaths passing through the same link are assigned to different wavelengths. Constraints (5) ensure that each lightpath traverses each link for at most once.

2.2 Related works

The RWA problem is a well-known combinatorial optimization problem that has been widely studied in the literature. The RWA has proven to be NP-complete in Chlamtac et al. (1992), and the authors provided the first greedy heuristic for the problem. The approaches and variants developed in the 1990s on the RWA problem were reviewed by Zang et al. (2000).

During past decades, various mathematical formulations have been developed for the RWA problem. The first path-based formulation for the RWA problem was developed by Ramaswami and Sivarajan (1995). Krishnaswamy and Sivarajan (2001) presented a mixed integer linear formulation which takes into account the maximum number of hops, among other logical and physical constraints, to minimize congestion. Dzongang et al. (2005) considered a variant of the RWA problem with the objective of minimizing the blocking rate.

A column generation algorithm for the RWA problem was developed by Lee et al. (2002). A very important milestone was established by Jaumard et al. (2009), who improved the column generation methods and obtained the optimum solution for several instances for the first time. The same authors also reviewed the existing column generation formulations in Jaumard et al. (2006, 2007).

Various heuristic and meta-heuristic algorithms have been proposed for solving the RWA problem. Some approaches decomposed the problem into two subproblems, the routing problem and the wavelength assignment problem, for example Banerjee and Mukherjee (1996) and Noronha and Ribeiro (2006), while others solved these two subproblems simultaneously, for example Manohar et al. (2002). Noronha and Ribeiro (2006) used a decomposition scheme in two distinct phases. Manohar et al. (2002) explored an alternative solution technique in the well-known maximum edge disjoint paths problem which can be naturally adapted to the RWA problem. The idea of bin packing heuristics was adapted to the min-RWA by Skorin-kapov (2007). The algorithm proposed by Skorin-Kapov was then improved by Noronha et al. (2008). Noronha et al also introduced a broader set of test instances for min-RWA. These instances contain random large scale problems which are considered as the most difficult ones, which are also used in this paper. Recently, the bin packing heuristics was embedded into a biased random-key genetic algorithm by Noronha et al. (2011). The genetic algorithm improved upon results from heuristics previously proposed in the literature. Martins et al. (2012) proposed a variable neighborhood descent algorithm with iterated local search for RWA. The authors tested their algorithm on the instances provided by Noronha et al. (2008). Results show that their algorithm are very competitive for solving most of the instances to the global optimum.

3 MN-ITS algorithm for min-RWA

The min-RWA can be solved from the point of view of constraint satisfaction by solving a series of RWA problems with a given number of wavelength k. Starting from an initial number of wavelength k which is large enough, our algorithm solves the k-RWA problem. As soon as the k-RWA problem is solved, we decrease k to $k - 1$ and solve again the k-RWA problem. This process is repeated until no legal k-RWA can be found. It is clear that a smaller k for a given instance leads to a harder k-RWA problem and our algorithm thus solves a series of problems of increasing difficulty.

In this section, we present our multi-neighborhood search approach for the k-RWA problem. Our proposed MN-ITS algorithm integrates several distinguishing features which ensure its effectiveness, including three complementary neighborhoods (denoted by N_c, N_r and N_e, see below), their combination and a unified incremental neighborhood evaluation method. These neighborhoods are explored in an alternating mode employing a rule that avoids certain neighboring solutions subject to the tabu conditions.

3.1 Search space and evaluation function

For a given k-RWA instance (G, L), where G represents the physical network and L represents the set of lightpaths, the proposed MN-ITS algorithm explores a search space Ω composed of all possible routes and wavelength assignments for each lightpath, i.e., $\Omega = \{X : X = (P, C)\}$ such that $p_l \in P$ represents the possible route for lightpath l in graph G and $c_l \in C$ represents the possible wavelength assigned to l. For any solution $X \in \Omega$, its quality is evaluated by the number of conflicts of this solution. The conflict is defined as the count number of common links shared by two lightpaths with a same wavelength. Therefore, a solution X with no conflicts corresponds to a legal solution for the k-RWA problem.

3.2 Main scheme

Iterated tabu search (ITS) is known to be a highly effective meta-heuristic framework for solving a large number of combinatorial optimization problems, which combines ILS with a tabu search (TS) algorithm (see Fu et al. (2013), Wang and Huang (2005), Lü and Huang (2009), Wang et al. (2002) and Wu et al. (2012)). Starting from an initial solution, a TS procedure is performed until a local optimum solution is reached. Then, the obtained local optimum solution is destructed by a perturbation operator and another round of TS is performed from the destructed solution. This process is repeated until a stopping criterion is met.

Our MN-ITS algorithm adapts an ITS algorithm in a multi-neighborhoods manner and its main scheme can be simply described in Algorithm 1. The best solution is kept in the memory during the search. At the end of each round, the procedure always perturbs the best solution to obtain a new start point for the next round. The stopping condition of our MN-ITS algorithm is the time limit.

Algorithm 1 The framework of the MN-ITS algorithm form k-RWA

```
1: procedure MN- ITS FOR K- RWA($X_0$)
2:     $X \leftarrow init\_kRWA(X_0)$
3:     $X_{best} \leftarrow X$
4:     repeat
5:         $X \leftarrow Tabu\_Search(X)$
6:         if $f(X) = 0$ then
7:             return $X$
8:         end if
9:         if $f(X) < f(X_{best})$ then
10:            $X_{best} \leftarrow X$
11:        end if
12:        $X \leftarrow Perturbation(X_{best})$
13:    until stop condition is met
14: end procedure
```

3.3 Initial solution

At the beginning of the MN-ITS algorithm, we use the BFD-RWA algorithm proposed in Skorin-kapov (2007) to generate a feasible solution using a large enough k (see Algorithm 2). The BFD-RWA algorithm is a bin packing based greedy heuristic algorithm, where wavelengths are treated as bins (w_c) and lightpaths as items (l). W and L represent the set of bins w_c and items l, respectively. w_c denotes a sub-graph of G, which is composed of all the unvisited links that can be used by the lightpaths assigned to wavelength c. Interested readers are referred to Skorin-kapov (2007) for more details.

Notice that for the k-RWA problem in our algorithm, the initial solution is inherited from the feasible solution X for $(k+1)$-RWA by removing the least used wavelength and randomly reassign the affected lightpaths with other wavelengths.

Algorithm 2 Initial solution generation method for RWA

```
1: procedure BFD- RWA($G, L$)
2:     Set $W \leftarrow \phi$ and $X \leftarrow \phi$
3:     for $l \in L$ do
4:         Find the bin $w_c \in W$ where there exists a feasible path for $l$.
5:         if no such bin exists then
6:             $w_c \leftarrow$ new copy of $G$
7:             $W \leftarrow W \cup \{w_c\}$
8:         end if
9:         Let $p$ be the shortest path between the endnotes of $l$ in $w_c$
10:          Assign the route $p$ and wavelength $c$ to lightpath $l$
11:          Delete edges in route $p$ from $w_c$
12:     end for
13:     return $X$
14: end procedure
```

3.4 Neighborhoods definition and exploration

In local search, a neighborhood is typically defined by a move operator mv, which transforms a solution X to generate a neighboring solution X', denoted by $X' = X \oplus mv$. Let $M(X)$ be the set of all possible moves which can be applied to X, then the neighborhood N of X is defined by $N(X) = \{X' : X' = X \oplus mv, mv \in M(X)\}$. The proposed MN-ITS algorithm jointly explores three neighborhoods which are defined by three move operators (denoted by mv_c, mv_r and mv_e).

3.4.1 Move operator mv_c

The move operator $mv_c(l)$ changes the wavelength of lightpath l to another one. In order to reduce the complexity of calculation, only lightpaths which produce conflicts are considered in this move operator. Then, the neighborhood defined by mv_c is given by $N_c(X) = \{\bar{X} : \bar{X} = X \oplus mv_c(l), l \in L, conflicts(l) > 0\}$.

Algorithm 3 Evaluation of neighborhood N_c

```
1: procedure EXPLORE $N_c$
2:     $\Delta_{best} = \infty$
3:     for all $\{l : l \in L;$ and $conflicts(l) > 0\}$ do
4:         $f_l = conflicts(l)$
5:         for all $c \in \{c : c \in C;$ and $c \neq c_l\}$ do
6:             $\Delta_{f_l} \leftarrow conflicts(l$ assigned to wavelength $c) - f_l$
7:             if $l$ is not tabu then
8:                 if $\Delta_{f_l} < \Delta_{best}$ then
9:                     $bestMove \leftarrow (l, c)$
10:                     $\Delta_{best} \leftarrow \Delta_{f_l}$
11:                 end if
12:             else
13:                 if $\Delta_{f_l} < \Delta_{best}$ and $f(X) + \Delta_{f_l} < F_{best}$ then
14:                     $bestMove \leftarrow (l, c)$
15:                     $\Delta_{best} \leftarrow \Delta_{f_l}$
16:                 end if
17:             end if
18:         end for
19:     end for
20:     return $bestMove$
21: end procedure
```

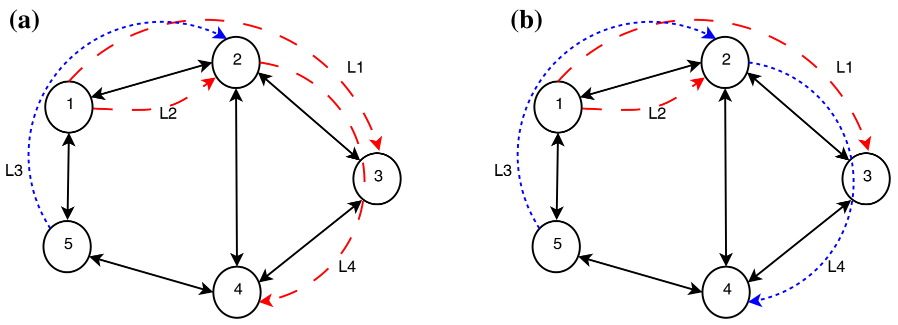

Fig. 1 An illustration of neighborhood move for N_c (Color figure online)

The evaluation of neighborhood N_c is described in Algorithm 3. For illustration, we consider an example as shown in Fig. 1. Assuming that there are four lightpaths and they are routed as shown in Fig. 1a, where lightpaths L1, L2, L4 are assigned to wavelength c_1 (the red dash lines) and lightpath L3 is assigned to wavelength c_2 (the blue dot line). One finds that lightpath L1 produces 2 conflicts and either lightpath L2 or L4 produces 1 conflict. It is straightforward that the best neighborhood move of N_c for the current solution is re-assigning lightpath L1 to c_2 or re-assigning lightpath L4 to c_2. The latter one is shown in Fig. 1b.

3.4.2 Move operator mv_r

The move operator $mv_r(l)$ changes the route of lightpath l to another candidate one. Like in Sect. 3.4.1, only lightpaths which produce conflicts are considered in this move operator. The neighborhood defined by mv_r is denoted as $N_r(X) = \{\bar{X} : \bar{X} = X \oplus mv_r(l), l \in L, conflicts(l) > 0\}$.

In this study, the evaluation of neighborhood is carried out by a Dijikstra-based shortest path algorithm, which can determine the route and examine the conflicts that it will produce at the same time. In this procedure, the candidate routes for each lightpath are evaluated and the best move for all the lightpaths is chosen for the next iteration.

For example, assuming that there are four lightpath requests like in Sect. 3.4.1, as shown in Fig. 2. In the current solution, lightpaths L1, L2 are assigned to wavelength c_1 (the red dash lines) and lightpaths L3, L4 are assigned to c_2 (the blue dot lines). Their routes are shown in Fig. 2a.

One finds that each of lightpaths L1 and L2 produces 1 conflict. A feasible solution can be obtained by changing the route of L1 to 1–5–4–3 or 1–5–4–2–3, or changing the route of L2 to 1–5–4–2. Each one can produce a conflict-free solution and the second candidate solution is shown in Fig. 2b.

The algorithm of evaluating the neighborhood moves for a certain lightpath is described in Algorithm 4. One observes that this procedure is almost the same as the improved Dijikstra's shortest path algorithm in Fredman and Tarjan (1984). The only difference with the Dijikstra's algorithm in Fredman and Tarjan (1984) lies in line 18 of Algorithm 4, where the conflict on the edge is its distance (weight). Thus,

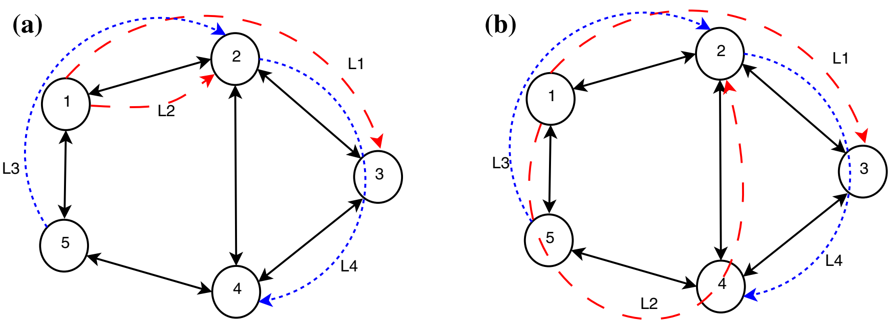

Fig. 2 The illustration of move in N_r (Color figure online)

applying this procedure on lightpath l, the path with the least conflict for l will be selected. Furthermore, according to our experiments, the length of lightpaths has also an effect on the solution quality. Usually, it is relatively easy to find a feasible solution if a lightpath has a short route. Therefore, we use a constant parameter ε (line 18 in Algorithm 4), which can help the search to obtain a balance between choosing a shorter and less conflict route, which is empirically set to 0.01. Thus, the algorithm will, under the circumstance of the least conflict, return a route as short as possible.

Algorithm 4 Find route for a certain lightpath

```
1: procedure FIND ROUTE FOR LIGHTPATH($l$)
2:     $same\_count \leftarrow 1$
3:     for $v \in V$ do
4:         if $v = l.source$ then
5:             $dist[v] \leftarrow \infty$
6:         else
7:             $dist[v] \leftarrow 0$
8:         end if
9:         $previous[v] \leftarrow undefined$
10:        $Q.add\_with\_priority(v, dist[v])$
11:    end for
12:    while $Q$ is not empty do
13:        $u \leftarrow Q.exact\_min()$
14:        if $u = l.sink$ then
15:            return GET_ROUTE($l$, $previous$)
16:        end if
17:        for each unvisited neighbor $v$ of $u$ do
18:            $alt \leftarrow dist[u] + conflict\_on(u, v) + \varepsilon$
19:            if $alt < dist[v]$ then
20:                $dist[v] \leftarrow alt$
21:                $previous[v] \leftarrow u$
22:                $Q.decrease\_priority(v, alt)$
23:            end if
24:        end for
25:    end while
26:    return GET_ROUTE($l$, $previous$)
27: end procedure
```

The data structure Q used in the algorithm is a Fibonacci heap, which is used to speed up the procedure (see Fredman and Tarjan (1984) for more details). According to Fredman and Tarjan (1984), the time complexity of the Dijikstra's algorithm using the

Fibonacci heap is $O(|E|+|V|log|V|)$. Since we have used almost the same algorithm here, the time complexity of this procedure is $O(|E|+|V|log|V|O(conflicts\_on))$, where $O(conflicts\_on)$ represents the time complexity of the procedure $conflict\_on$ in line 18. If the conflict value on each edge is stored in a memory and updated after each iteration, the time complexity of $conflict\_on$ will be $O(1)$. Thus, the time complexity of Algorithm 4 is also $O(|E|+|V|log|V|)$.

3.4.3 Move operator mv_e

The move operator mv_e, which defines neighborhood N_e, consists of a sequence of moves involving changing both wavelength and route. It can be described as follows:

1. Randomly choose a conflicting lightpath l_1 and let c_o be the wavelength of l_1. Temporarily remove l_1 from the current solution.
2. Find a lightpath l_i whose conflicts can be eliminated if its wavelength is changed to c_o with a new route.
3. Change the wavelength of l_i to c_o and assign the new route to it.
4. Let c_o be the former wavelength of l_i.
5. Repeat steps 2–4 until there is no eligible lightpath.
6. Restore lightpath l_1 (remain its route unchanged) and assign its wavelength to c_o

This procedure is quite similar to the ejection chain neighborhood in Glover (1996), which has been widely used in many applications (Gonzalez-Velárde and Laguna 2002).

Consider an example as shown in Fig. 3. There are 7 lightpaths assigned to two different wavelengths, which are shown as the dot blue lines and the dash red lines in Fig. 3[Step 1]. One observes that the conflict value of this solution is 2, on fiber C–B and B–E, respectively. The move operator mv_e addressing this situation is as follows:

1. Choose the lightpath traversing A–B–E to be l_1 (Fig. 3[Step 1]). Set c_o to be the wavelength of l_1, which is blue in this case. and remove l_1 from the network temporarily.
2. Find a lightpath l_2 whose conflicts can be eliminated if it is assigned with wavelength c_o (blue) and a new route. Here we find that changing lightpath (l_2) traversing D–C–B to D–E–B will meet the requirement. So we apply this move (Fig. 3[Step 2]–[Step 3]). Also, c_o will be changed to the previous wavelength (red) of l_2.
3. At this point, there is no more eligible lightpath. So the procedure stops. We restore l_1 to the network and reassign its wavelength to c_o.

In this case the total conflict value is reduced after these moves. However, this is not always the case. Although each iteration in move mv_e aims at reducing the conflicts, the total conflicts may still increase because of the last step. In addition, it is possible that there does not exist an available lightpath l_i in step 2 of the above procedure. In this case, the procedure just restores the removed lightpath l_1 in the first step and does not proceed any move.

The significance of the move mv_e lies in the fact that it helps improving the objective value when the number of conflicts is relatively large, and provides a diversification mechanism when the local optimum trap for mv_c and mv_r is reached. One can consider the move operators mv_c and mv_r as an intensification oriented phase, while the

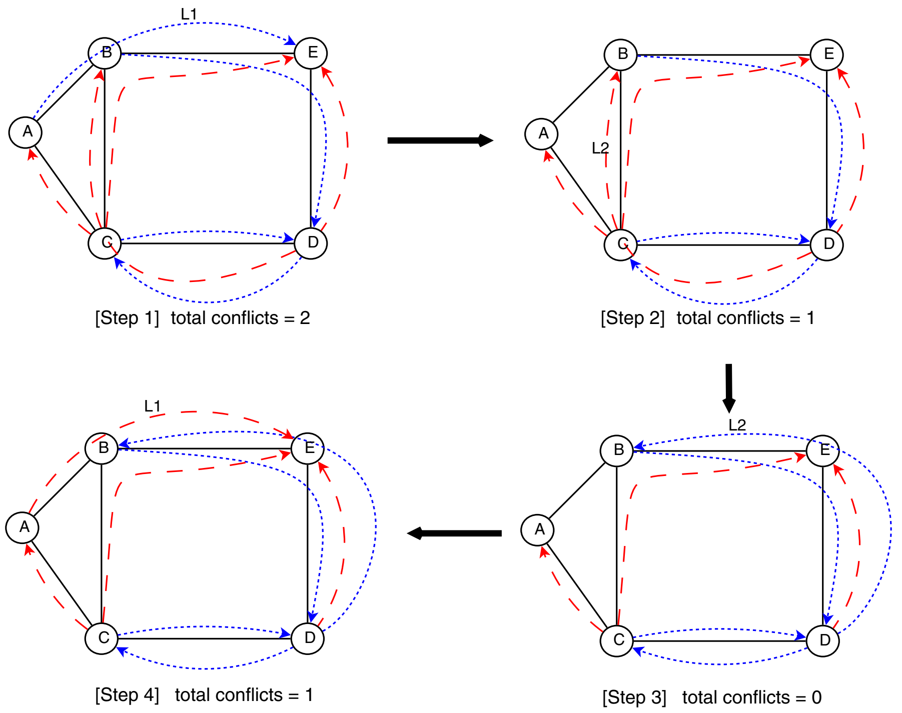
L1
A
B
C
D
E
[Step 1] total conflicts = 2
L2
[Step 2] total conflicts = 1
[Step 3] total conflicts = 0
[Step 4] total conflicts = 1

Fig. 3 An illustration for neighborhood N_e (Color figure online)

move operator mv_e is a diversification oriented mechanism which helps the search to enhance its search efficiency and effectiveness. The significance of using these three neighborhoods will be illustrated in Sect. 4.4.

The operator mv_e is performed in an iterative way. At each iteration, Algorithm 4 is used in step 2 to find the route for each lightpath, and the worst case is to evaluate all the lightpaths. Therefore, the time complexity for each iteration of mv_e is $O(|L|(|E| + |V|log|V|))$. Usually, the procedure will be executed for several times. Thus, the time complexity of evaluating the whole operator mv_e will be a constant time of $O(|L|(|E|+|V|log|V|))$. Although the time complexity of the operator mv_e is higher than that of mv_c and mv_r, it will not slow down the search too much. The reason lies in the fact that the execution condition for mv_e is rather strict such that it will be only performed in very rare situations.

3.4.4 Incremental evaluation

Given a neighborhood structure, it is particularly important to be able to rapidly determine the effect of a move on the objective function $f(X)$. For this purpose, we propose a fast incremental evaluation technique to evaluate the Δ values. The proposed MN-ITS algorithm maintains a $|C| \times |E|$ matrix T from which the move values for all the possible moves can be rapidly calculated. $T[c][e]$ measures the number of lightpaths which are assigned to wavelength c and traverse link e. It can be calculated as:

$$T[c][e] = |\{l : c_l = c \text{ and } e \in p_l\}|$$

where c_l represents the wavelength of lightpath l and p_l represents the route of l.

Note that this matrix is only initialized at the beginning of each MN-ITS procedure and is updated in a fast manner at each local search iteration. The move operators mv_c and mv_r can be considered as the special cases of a general move $< l, c', c^*, p', p^* >$, which represents changing the wavelength of lightpath l from c' to c^* and changing the route of l from p' to p^*. Thus, the move operator mv_e can be looked as a sequence of moves $< l, c', c^*, p', p^* >$. Therefore, the incremental evaluation of all the neighborhood moves in the proposed MN-ITS algorithm are the same as move $< l, c', c^*, p', p^* >$, whose incremental value can be determined as:

$$\Delta f(X)_{<l,c',c^*,p',p^*>} = \sum_{e \in p^*} min(T^*[c^*][e], 1) - \sum_{e \in p'} min(T[c'][e] - 1, 1)$$

where

$$T^*[c][e] = \begin{cases} T[c][e] - 1, & \text{for } e \in p'_l \\ T[c][e], & \text{for } e \notin p'_l \end{cases}$$

We now describe how the matrix T is updated once a move is performed during the MN-ITS local search procedure. If a move $< l, c', c^*, p', p^* >$ is performed, only the elements affected by this move need to be updated, which are the elements in set $ST = \{T[c][e] : (c = c' \text{ or } c = c^*) \text{ and } e \in p' \cup p^*\}$.

In fact, since only one lightpath is changed in a neighborhood move $< l, c', c*, p', p^* >$, the value of these affected elements can be simply updated as follows, rather than being re-calculated from scratch:

$$T[c'][e] \leftarrow T[c'][e] - 1, \ e \in p'$$
$$T[c^*][e] \leftarrow T[c^*][e] + 1, \ e \in p^*$$

Usually, there are very few links involved in one route for a lightpath. Thus, the number of updated elements would be very few.

3.5 Tabu search procedure

Our local search procedure exploits the three neighborhoods (N_c, N_r and N_e) in an alternating way. More precisely, we start the algorithm with one neighborhood. When one move is applied, we switch to another neighborhood. This process is repeated until the procedure meet the termination condition. In our case, the algorithm begins with the basic neighborhood $N_c : N_c \rightarrow N_r \rightarrow N_e \rightarrow N_c \rightarrow N_r \rightarrow N_e \ldots$.

In our local search procedure, each neighborhood is not completely searched until a local optimum solution is reached. Instead, we apply one move for each neighborhood before proceeding to the next neighborhood. Since the three neighborhoods use a unified incremental evaluation method, the switch is very natural.

At each iteration of the local search procedure, the best improvement strategy is applied for neighborhoods N_c and M_r, while the first improvement strategy is employed for neighborhood N_e. In spite of the unified incremental evaluation method and the tabu mechanism, the three neighborhoods are independent among each other. Thus, it is possible that a same lightpath may be used in different neighborhoods.

The tabu search mechanism is implemented in the proposed MN-ITS algorithm to forbid the previously visited solutions to be revisited (Glover (1989, 1990). In this work, when a move involves changing the wavelength of lightpath l from c_i to another wavelength, lightpath l is forbidden to accept wavelength c_i until lightpath l changes its routing path. Otherwise, assigning lightpath l with wavelength c_i is declared tabu for a certain time span (tt iterations). That is to say, the tabutenure of our tabu list is a dynamic value depending on the search history.

Since the tabu list records the attributes of a solution instead of the solutions themselves, it is possible that a candidate solution in the tabu list could lead to a solution better than the best solution found so far. Therefore, in our MN-ITS algorithm, a simple aspiration criterion is applied which allows the best move in the neighborhood to be performed in spite of being tabu if it leads to a solution better than the current best solution.

The tabu search phase stops when the current best solution cannot be improved within a given number of iterations which is called the threshold of the tabu search and is denoted as T. Algorithm 5 illustrates the pseudo-code of the tabu search phase.

Algorithm 5 Tabu search

```
1: procedure TABU_SEARCH($l$)
2:     $F_b \leftarrow f(X)$
3:     $fail\_count \leftarrow 0$
4:     while $fail\_count < T$ do
5:         $mv_c \leftarrow Explore\_N_c()$
6:         Apply $mv_c$
7:         Set tabu status for the lightpath in $mv_c$
8:         $mv_r \leftarrow Explore\_N_r()$
9:         Apply $mv_r$
10:        Relieve the tabu status for the lightpath in $mv_r$
11:        $Explore\_N_e()$
12:        if $f(X) \geq F_b$ then
13:            $fail\_count \leftarrow fail\_count + 1$
14:        else
15:            $F_b \leftarrow f(X)$
16:            $fail\_count \leftarrow 0$
17:        end if
18:    end while
19:    return $X$
20: end procedure
```

The MN-ITS algorithm proposed in this paper is very similar to the variable neighborhood decent (VND) algorithm proposed in Martins et al. (2012). However, there are obvious differences. The main difference lies in the neighborhood definition. The proposed MN-ITS algorithm converts the original min-RWA problem to a series of decision problems (k-RWA). In each k-RWA procedure, MN-ITS attempts to change an infeasible solution in order to satisfy all the constraints. On the contrary, the VND algorithm proposed in Martins et al. (2012) directly solves the min-RWA problem. Moreover, the neighborhood switching strategies are different between the two algorithms. The idea of our proposed MN-ITS algorithm provides another perspective in designing meta-heuristics for solving the min-RWA algorithm compared with the VND algorithm.

3.6 Perturbation operator

The perturbation phase in a meta-heuristic algorithm usually plays an important role in diversifying the search when the search gets stuck in local optimum trap (see Huang and Ye (2011), Ye et al. (2011)). In this paper, when the tabu search phase terminates, we employ a perturbation operator to destruct the reached local optimum solution in order to jump out of this local optimum trap and explore a more promising region of the search space. For the min-RWA problem, the perturbation operator employed in our algorithm can be described in Algorithm 6.

Algorithm 6 Perturbation

```
1: procedure PERTURBATION($X$)
2:   for Each wavelength $c \in C$ do
3:     if There exists a conflicting lightpath $l$ assigned to $c$ then
4:       Randomly choose a wavelength $c' \in C$ apart from $c$
5:       if There exists a conflicting lightpath $l'$ assigned to $c'$ then
6:         Exchange the wavelengths of $l$ and $l'$
7:       else
8:         Assign lightpath $l$ to wavelength $c'$
9:       end if
10:    end if
11:  end for
12:  return $X$
13: end procedure
```

For each wavelength c, randomly choose a conflicting lightpath if there is any. Then randomly choose another wavelength c'. If there exists a conflicting lightpath assigned to c', then exchange the wavelengths of these two lightpaths. Otherwise, assign the lightpath to wavelength c'.

Usually, a too strong perturbation will behavior like a multi-start method. So, the perturbation used here is a relatively small perturbation such that the perturbed solution is not very far from the previous local optimum solution. Notice that we use a different move from the three neighborhood moves which allows our algorithm to explore new search regions to some extent. Our above perturbation operator performs just once after each local search phase, and a stronger perturbation can be obtained by repeating this operator for several times. However, according to our experiments, just performing this operator for once is good enough for our algorithm, while stronger perturbation will diversify the search too much, which can be justified in Sect. 4.2.

4 Computational results

In this section, experiments are carried out to evaluate the performance of the proposed MN-ITS algorithm.

4.1 Test instances

To ensure a fair comparison, we test our MN-ITS algorithm on the same two sets of RWA problem instances as those used by the reference algorithms in the literature. Set W is a collection of the most widely studied real world instances.[1] Set Y instances were introduced by Noronha et al. (2008), which are randomly generated with 100 nodes. The probability that there is a link between a pair of nodes is equal to 0.03, 0.04, and 0.05, respectively, and the probability that there is a lightpath request between a pair of nodes is equal to 0.2, 0.4, 0.6, 0.8, and 1.0, respectively. Combining each possible value of the probability of the link and the probability of the lightpath requests, totally 15 groups of instances are generated, each with 5 instances.

The lightpath requests for both sets of instances are asymmetric. The LB column in each table represents the lower bound for each instance. The lower bound can be obtained by relaxing the wavelength continuity constraints and relaxing the integer restriction of the flow decision variables (Banerjee and Mukherjee 1996).

4.2 Calibration

In this section, we conduct an experiment to determine the appropriate values of some important parameters of the MN-ITS algorithm.

- The tabu tenure (tt). The value tt is defined as a constant value in the MN-ITS. Three values, 2, 5 and 10, are tested.
- The threshold T used to determine whether the algorithm is trapped in the local optimum trap. Three values, 600, 1000 and 1500, are considered.
- The perturbation strength. The perturbation strength in the MN-ITS is recognized as the times that the perturbation phase is executed. Two values, 1 and 5, are tested.

Calibration experiments are conducted on several representative instances in Y set with linking density of 0.03. The experiments record the best solution value that the MN-ITS can obtain in 5 min. Each run is repeated for 10 times with different random seeds. Table 1 displays the summarized results for each parameter combination. The first three columns represent the parameter settings, and the last column presents the average gaps to the lower bounds.

From Table 1, one finds that the perturbation strength of 5 diversifies the search too much and it obtains relatively worse results than the strength of 1. The differences between other parameter settings are not so much. However, one observes that the 3rd

[1] Which are available on the web site: http://mauricio.resende.info/data/rwa.tar.gz.

Table 1 Calibration

tt	*T*	Strength	Gap (%)
2	600	1	3.52
2	600	5	9.23
2	1000	1	3.37
2	1000	5	6.26
2	1500	1	3.37
2	1500	5	8.26
5	600	1	4.74
5	600	5	9.23
5	1000	1	3.52
5	1000	5	8.26
5	1500	1	3.52
5	1500	5	8.26
10	600	1	4.74
10	600	5	9.02
10	1000	1	4.74
10	1000	5	9.23
10	1500	1	3.53
10	1500	5	9.02

Table 2 Comparison with the reference algorithms on set W instances

Instance	LB	MN-ITS		MS-BFD		VND	
		Obj	Gap (%)	Obj	Gap (%)	Obj	Gap (%)
ATT	20	*20*	0.00	25	25.00	*20*	0.00
ATT2	113	*113*	0.00	*113*	0.00	*113*	0.00
Finland	46	*46*	0.00	47	2.17	*46*	0.00
NSF.1	22	*22*	0.00	23	4.55	*22*	0.00
NSF.3	22	*22*	0.00	*22*	0.00	*22*	0.00
NSF.12	38	*38*	0.00	39	2.63	*38*	0.00
NSF.48	41	*41*	0.00	*41*	0.00	*41*	0.00
NSF2.1	21	*21*	0.00	*21*	0.00	*21*	0.00
NSF2.3	21	*21*	0.00	*21*	0.00	*21*	0.00
NSF2.12	35	*35*	0.00	*35*	0.00	*35*	0.00
NSF2.48	39	*39*	0.00	*39*	0.00	*39*	0.00
brasil	48	*48*	0.00	*48*	0.00	*48*	0.00
EON	22	*22*	0.00	*22*	0.00	*22*	0.00
Average			0.00		0.74		0.00

Table 3 Comparison on set Y3 instances

Instance	LB	MN-ITS		MS-BFD		VND	
		Obj	Gap (%)	Obj	Gap (%)	Obj	Gap (%)
Y.3.20.1	27	*29*	7.41	33	22.22	*29*	7.41
Y.3.20.2	33	*33*	0.00	*33*	0.00	*33*	0.00
Y.3.20.3	29	*29*	0.00	31	6.90	*29*	0.00
Y.3.20.4	26	*28*	7.69	31	19.23	*28*	7.69
Y.3.20.5	28	29	3.57	31	10.71	**28**	2.86
Y.3.40.1	53	*57*	7.55	62	18.49	*57*	7.55
Y.3.40.2	59	*59*	0.00	*59*	0.00	*59*	0.00
Y.3.40.3	61	*61*	0.00	*61*	0.00	*61*	0.00
Y.3.40.4	50	*54*	10.00	58	16.00	*54*	10.00
Y.3.40.5	53	*56*	5.66	60	13.21	*56*	5.66
Y.3.60.1	81	**86**	6.17	93	14.81	87	7.41
Y.3.60.2	89	*89*	0.00	*89*	0.00	*89*	0.00
Y.3.60.3	91	*91*	0.00	*91*	0.00	*91*	0.00
Y.3.60.4	78	*80*	2.56	85	9.23	*80*	2.56
Y.3.60.5	77	*82*	6.49	86	12.73	*82*	6.49
Y.3.80.1	106	**114**	7.55	123	16.04	115	9.25
Y.3.80.2	117	*117*	0.00	*117*	0.00	*117*	0.00
Y.3.80.3	118	*118*	0.00	*118*	0.17	*118*	0.00
Y.3.80.4	105	*106*	0.95	112	6.67	*106*	0.95
Y.3.80.5	104	*109*	4.81	117	9.62	*109*	4.81
Y.3.100.1	131	**141**	7.63	151	15.27	143	9.31
Y.3.100.2	146	*146*	0.00	*146*	0.00	*146*	0.00
Y.3.100.3	146	*146*	0.00	*146*	0.00	*146*	0.00
Y.3.100.4	131	*132*	0.76	138	5.80	*132*	0.76
Y.3.100.5	129	*136*	5.43	141	9.30	*136*	5.43
Average			3.37		8.26		3.53

and 5th parameter settings lead to the best performance. In the following, we use the 3rd parameter setting (2, 1000, 1) in the MN-ITS.

4.3 Experimental protocol

Our algorithm is programmed in C++ and the experiments are conducted on a PC with an intel i3 CPU of 2.1GHz and 4 GB of RAM. The results are compared with three reference algorithms: the Multistart-BFD algorithm and GA-RWA proposed by Noronha et al. (2011), and the VDN-ILS algorithm proposed by Martins et al. (2012). 5 independent runs are performed for each instance with the time limit of 5 min for each run, and the best objective values are recorded. The

Table 4 Comparison on set Y4 instances

Instance	LB	MN-ITS		MS-BFD		VND	
		Obj	Gap (%)	Obj	Gap (%)	Obj	Gap (%)
Y.4.20.1	17	*19*	11.76	21	23.53	*19*	11.76
Y.4.20.2	28	*28*	0.00	*28*	0.00	*28*	0.00
Y.4.20.3	23	*23*	0.00	*23*	0.00	*23*	0.00
Y.4.20.4	19	*19*	0.00	*20*	8.42	*19*	0.00
Y.4.20.5	17	*19*	11.76	21	23.53	*19*	11.76
Y.4.40.1	31	*35*	12.90	38	22.58	*35*	12.90
Y.4.40.2	57	*57*	0.00	*57*	0.00	*57*	0.00
Y.4.40.3	43	*43*	0.00	*43*	0.00	*43*	0.00
Y.4.40.4	38	*38*	0.00	*38*	0.00	*38*	0.00
Y.4.40.5	33	*37*	12.12	40	21.21	*37*	12.12
Y.4.60.1	47	*53*	12.77	56	19.15	*53*	12.77
Y.4.60.2	86	*86*	0.00	*86*	0.00	*86*	0.00
Y.4.60.3	64	*64*	0.00	*64*	0.00	*64*	0.00
Y.4.60.4	58	*58*	0.00	*58*	0.00	*58*	0.00
Y.4.60.5	49	*55*	12.24	58	19.18	*55*	12.24
Y.4.80.1	62*	**69**	11.29	73	17.74	70	12.90
Y.4.80.2	118	*118*	0.00	*118*	0.00	*118*	0.00
Y.4.80.3	81	*81*	0.00	*81*	0.00	*81*	0.00
Y.4.80.4	78	*78*	0.00	*78*	0.00	*78*	0.00
Y.4.80.5	65	**71**	9.23	76	16.29	72	10.77
Y.4.100.1	76	*86*	13.16	91	19.74	*86*	13.16
Y.4.100.2	146	*146*	0.00	*146*	0.00	*146*	0.00
Y.4.100.3	98	*98*	0.00	*98*	0.00	*98*	0.00
Y.4.100.4	98	*98*	0.00	*98*	0.00	*98*	0.00
Y.4.100.5	80	*89*	11.25	93	16.50	*89*	11.75
Average			4.74		8.34		4.93

* According to Noronha et al. (2011) and Martins et al. (2012), the lower bound of this instance is 47. However, according to our experiment, it should be 62

statistics for the reference algorithms are obtained from reference Martins et al. (2012).

4.4 Comparison with the reference algorithms

The computational results for set W instances are shown in Table 2. Column LB gives the lower bounds of each instance. Column Obj reports the best result obtained by each reference algorithm for each instance and column Gap is calculated as $(Obj - LB)/LB \times 100\,\%$. Last row presents the average gaps to the lower bounds over all the tested instances. For illustrative purpose, better results are indicated in

Table 5 Comparison on set Y5 instances

Instance	*LB*	MN-ITS		MS-BFD		VND	
		Obj	Gap (%)	Obj	Gap (%)	Obj	Gap (%)
Y.5.20.1	13	*13*	0.00	14	7.69	*13*	0.00
Y.5.20.2	17	*17*	0.00	*17*	0.00	*17*	0.00
Y.5.20.3	12	13	8.33	13	8.33	**12**	5.00
Y.5.20.4	17	*17*	0.00	*17*	0.00	*17*	0.00
Y.5.20.5	15	*15*	0.00	*15*	0.00	*15*	0.00
Y.5.40.1	24	*24*	0.00	25	4.17	*24*	0.00
Y.5.40.2	31	*31*	0.00	*31*	0.00	*31*	0.00
Y.5.40.3	22	*23*	4.55	24	9.09	*23*	4.55
Y.5.40.4	33	*33*	0.00	*33*	0.00	*33*	0.00
Y.5.40.5	28	*28*	0.00	*28*	0.00	*28*	0.00
Y.5.60.1	33	*35*	6.06	36	11.52	*35*	6.06
Y.5.60.2	45	*45*	0.00	*45*	0.00	*45*	0.00
Y.5.60.3	34	*34*	0.00	35	2.94	*34*	0.00
Y.5.60.4	48	*48*	0.00	*48*	0.00	*48*	0.00
Y.5.60.5	40	*40*	0.00	*40*	0.00	*40*	0.00
Y.5.80.1	43	47	9.30	47	11.16	**46**	6.98
Y.5.80.2	59	*59*	0.00	60	1.69	*59*	0.00
Y.5.80.3	43	45	4.65	45	4.65	**44**	2.33
Y.5.80.4	63	*63*	0.00	*63*	0.00	*63*	0.00
Y.5.80.5	53	*53*	0.00	*53*	0.00	*53*	0.00
Y.5.100.1	55	*57*	3.64	59	7.27	*57*	3.64
Y.5.100.2	73	74	1.37	75	2.74	**73**	0.00
Y.5.100.3	53	55	3.77	56	5.66	**54**	3.40
Y.5.100.4	77	*77*	0.00	77	0.00	*77*	0.00
Y.5.100.5	66	*66*	0.00	*66*	0.00	*66*	0.00
Average			1.67		3.18		1.30

bold, while equal best ones are indicated in italic. (Tables 3, 4 and 5 adopt the same criteria).

It can be observed from the table that the proposed MN-ITS algorithm hits the lower bounds for all the instances in this set, implying that our algorithm can reach the optimum solutions for all the instances in set W.

Tables 3, 4 and 5 respectively present the results for set Y instances with network density of 0.03, 0.04 and 0.05. From Tables 3, 4 and 5, one observes that the proposed MN-ITS algorithm hits the lower bounds for 42 out of the 75 instances. Our MN-ITS algorithm is able to yield better or equal results to MS-BFD for all the instances. Comparing with the VND algorithm, the proposed MN-ITS algorithm obtains equal results for most instances. Particularly, for 5 instances, the proposed MN-ITS algorithm obtains better results than those obtained by VND,

Table 6 Comparison with GA-RWA on a set of difficult instances

Instance	LB	MN-ITS		GA-RWA	
		Obj	Gap (%)	Obj	Gap (%)
ATT	20	**20**	0.00	24	20.00
ATT2	113	113	0.00	113	0.00
Finland	46	46	0.00	46	0.00
NSF.3	22	22	0.00	22	0.00
NSF.12	38	**38**	0.00	39	2.63
Y.3.40.5	53	**56**	5.66	59	11.32
Y.3.60.5	77	**82**	6.49	86	11.69
Y.3.80.1	106	**114**	7.55	122	15.05
Y.3.80.5	104	**109**	4.81	113	8.65
Y.4.20.4	19	**19**	11.76	20	5.26
Y.4.60.5	49	**55**	12.24	58	18.37
Y.4.80.1	62	**69**	11.29	73	17.74
Y.4.80.5	65	**71**	9.23	75	15.38
Y.4.100.1	76	**86**	13.16	90	18.42
Y.5.60.1	33	**35**	6.06	36	9.09
Y.5.80.1	43	47	9.30	47	9.30
Y.5.80.2	59	**59**	0.00	60	1.69
Y.5.100.1	55	**57**	3.64	58	5.45
Y.5.100.2	73	74	1.37	74	1.37
Average			5.49		9.02

which are Y.3.60.2, Y3.80.1, Y.3.100.1, Y.4.80.1 and Y.4.80.5. However, for 6 instances Y.3.20.5, Y5.20.3, Y.5.80.1, Y.5.80.3, Y.5.100.2 and Y.5.100.3, the MN-ITS obtains worse results than VND. One finds that 5 out of these 6 instances are in set Y5. This indicates that our MN-ITS algorithm has strong capability to solve the RWA problems with sparse density, while it performs slightly worse for dense instances.

Table 6 presents a comparison between the MN-ITS and GA-RWA on the subset of the previously tested instances which are considered as the hardest ones by Noronha et al. (2011). Column Obj represents the best objective that the corresponding algorithm can obtain. The bold value indicates that the corresponding algorithm gets better results than the other one. One finds that the MN-ITS algorithm yields better results than GA-RWA for most of these hard instances, demonstrating the superiority of the MN-ITS in terms of the solution quality.

5 Analysis and discussion

Now we turn our attention to analyzing some important features of the MN-ITS algorithm, including the importance of the proposed incremental evaluation technique and the influence of the combined neighborhoods.

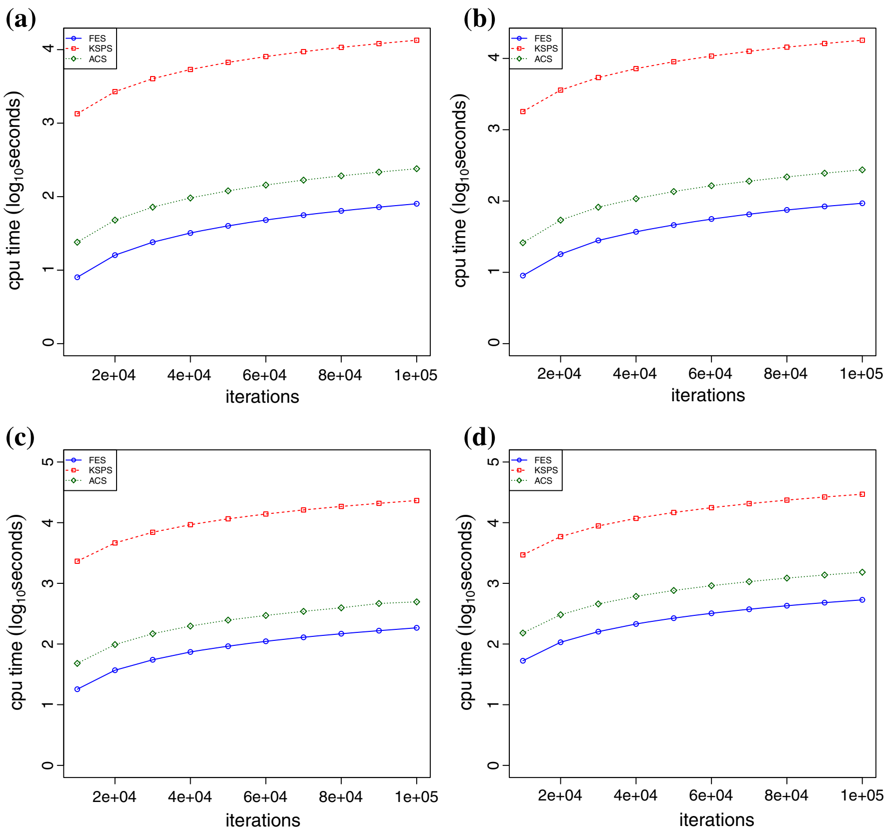

Fig. 4 Performance comparison of MN-ITS algorithm with and without the fast evaluation technique (Color figure online)

5.1 Importance of the incremental evaluation technique

For a local search method, it is particularly important to be able to rapidly determine the effect of a move on the objective function value. As described in Sect. 3.4.2, we proposed a tree search algorithm to accelerate neighborhood evaluation. Furthermore, we also proposed a unified incremental evaluation technique in Sect. 3.4.4 for evaluating the three neighborhood moves, where a $|C| \times |E|$ matrix T is maintained to help to calculate the move values of all possible candidate moves. Once a move is performed, only the values of matrix T affected by the move are accordingly updated.

In order to evaluate the effectiveness of the incremental evaluation techniques, we carry out computational experiments to compare the performance of our MN-ITS algorithm with and without this technique on several representative instances: Y.3.40.2, Y.3.60.3, Y.3.80.4 and Y.3.100.5. Note that similar results can be observed on other instances. The following three strategies are considered: Our fast incremental updating strategy used in this paper, called fast evaluation strategy (FES); The N_r and N_e are evaluated using k-shortest path algorithm rather than the tree search method,

Table 7 Comparison of neighborhoods

Instance	Obj	$N_c + N_r + N_e$	$N_c + N_r$	$N_c + N_e$
Y.3.20.1	33	1.3	9.3	6.2
	32	15.2	61.1	149.5
	31	95.4	232.1	–
Y.3.40.4	60	0.6	2.2	4.2
	59	1.1	29.1	84.4
	58	19.7	79.5	–
	57	66.8	149.2	–
Y.3.60.5	87	2.6	3.9	7.9
	86	12.8	207.3	126.4
	85	297.5	–	–
Y.3.80.1	125	0.5	12.2	5.4
	124	11.4	89.7	119.1
	123	23.4	224.2	254.6
	122	90.3	–	–
	121	174.9	–	–
Y.3.100.5	145	1.1	1.5	4.0
	144	10.6	24.9	–
	143	121.0	104.4	–
	142	169.0	–	–
	141	251.5	–	–

Table 8 Comparison of neighborhoods

Instance	Obj	$N_c + N_r + N_e$	$N_c + N_r$	$N_c + N_e$
Y.4.20.5	21	0.5	6.8	17.5
	20	35.1	92.6	–
Y.4.40.5	39	14.3	65.6	30.2
	38	243.6	–	–
Y.4.60.1	57	0.3	0.2	1.2
	56	13.6	62.3	–
	55	72.2	–	–
Y.4.80.1	73	5.6	46.6	51.7
	72	51.8	–	–
Y.4.100.5	92	18.4	69.8	87.8
	91	207.6	–	–

called k-shortest path strategy (KSPS); The fast incremental evaluation technique is not used and the neighborhood moves are calculated from scratch using the tree search method, called all calculating strategy (ACS).

Table 9 Comparison of neighborhoods

Instance	Obj	$N_c + N_r + N_e$	$N_c + N_r$	$N_c + N_e$
Y.5.20.1	15	2.1	20.7	1.9
	14	29.3	88.7	199.5
Y.5.40.3	26	6.7	13.0	10.0
	25	8.9	130.0	–
	24	54.9	–	–
Y.5.60.1	38	1.5	144.9	20.0
	37	48.5	–	–
Y.5.80.4	64	4.5	78.2	52.0
	63	129.7	–	–
Y.5.100.2	81	5.5	24.1	53.3
	80	8.8	176.4	–
	79	11.1	–	–
	78	96.7	–	–
	77	135.1	–	–
	76	200.4	–	–

For these instances, MN-ITS is independently run for five times. The evolution of the average CPU time with the number of iterations is shown in Fig. 4. In order to show the differences more clearly, the values shown in the figures are the base 10 logarithm of the CPU time measured in seconds.

One finds that the average CPU time of MN-ITS with the FES is much faster than other two strategies KSPS and ACS. For example, for instance Y.3.30.2, the FES is respectively about 70 and three times faster than KSPS and ACS. This experiment clearly demonstrates the importance of the incremental evaluation technique proposed in this paper.

5.2 Importance of the neighborhoods combination

In this section, we attempt to figure out which neighborhood is the most essential one in our MN-ITS algorithm. The experiments are performed by comparing the original MN-ITS algorithm and several simplified versions of our MN-ITS algorithm. Specifically, we consider two variants of the MN-ITS algorithm respectively with neighborhoods N_c and N_r, as well as the algorithm with neighborhoods N_c and N_e.

Tables 7, 8 and 9 show the results of the above mentioned variants of MN-ITS for several typical instances. Column Obj reports the objective values obtained by the MN-ITS in 5 min. Other columns gives the seconds used to reach the Obj value with the corresponding variants. The dash in the tables means that the algorithm fails to obtain this Obj value within the time limit. Here, each variant of the algorithm includes neighborhood N_c, because N_c is the basic neighborhood of our algorithm, without which our MN-ITS algorithm works very poorly. Thus, we do not report its results here.

From these tables, one finds that the original MN-ITS algorithm outperforms other variants. This indicates that each neighborhood is essential in the proposed MN-ITS algorithm. The algorithm without neighborhood N_r is the worst in terms of both solution quality and time efficiency according to our experiments. In some cases, the variants with neighborhood $N_c + N_r$ can obtain the same objective values as the original MN-ITS, but it needs much more time to obtain the same results.

From this experiment, one observes that each neighborhood is important in our MN-ITS algorithm. The combination of these different neighborhoods make the algorithm much more powerful and each neighborhood is meaningful in the proposed MN-ITS algorithm.

6 Conclusions and future work

The RWA problem, which has proven to be NP-hard, is widely studied in both academia and industry. In this paper, we have presented a multi-neighborhood based iterated tabu search algorithm for tackling this challenging problem.

We test the MN-ITS algorithm on 88 instances widely used in the literature, and compared the results with other two state-of-the-art algorithms. The results show that the MN-ITS algorithm matches the optimal solutions for all the real world instances and hits the lower bounds for 42 instances in the set Y benchmarks. Moreover, our MN-ITS algorithm is able to find better upper bounds for 5 instances, but gets worse results than the best upper bounds for 6 instances.

Our further analysis indicates that the unified fast incremental evaluation technique proposed in this paper is essential to enhance the efficiency of our MN-ITS algorithm. In addition, all the three neighborhoods are essential for affecting the effectiveness and efficiency of the MN-ITS algorithm.

There are several directions to extend this work. One immediate possibility is to incorporate the present TS algorithm as well as an appropriate crossover operator into the framework of an evolutionary algorithm. The other possibility is to develop new perturbation operators that consider more problem specific knowledge to enhance the performance of the present MN-ITS algorithm.

Acknowledgments We thank the anonymous referees for their helpful comments to improve our paper significantly. This paper is dedicated to memorizing the authors' mentor Prof. Wenqi Huang for his lifelong instruction. This research was supported in part by the National Natural Science Foundation of China under Grant Number 61370183 and the program for New Century Excellent Talents in University (NCET 2013).

References

Banerjee D, Mukherjee B (1996a) A practical approach for routing and wavelength assignment in large wavelength-routed optical networks. IEEE J Sel Areas Commun 14(5):903–908

Chlamtac I, Ganz A, Karmi G (1992) Lightpath communications: an approach to high bandwidth optical wan's. IEEE Trans Commun 40(7):1171–1182

Dutta R, Rouskas GN (2000) A survey of virtual topology design algorithms for wavelength routed optical networks. Opt Netw Mag 1(1):73–89

Dzongang C, Galinier P, Pierre S (2005) A tabu search heuristic for the routing and wavelength assignment problem in optical networks. IEEE Commun Lett 9(5):426–428

Fredman ML, Tarjan R (1984) Fibonacci heaps and their uses in improved network optimization algorithms. In: 25th Annual Symposium on Foundations of Computer Science, 1984, pp. 338–346

Fu Z, Huang W, Lü Z (2013) Iterated tabu search for the circular open dimension problem. Eur J Oper Res 225(2):236–243

Glover F (1989) Tabu search part i. ORSA J Comput 1(3):190–206

Glover F (1990) Tabu search part ii. ORSA J Comput 2(1):4–32

Glover F (1996) Ejection chains, reference structures and alternating path methods for traveling salesman problems. Discret Appl Math 67(1–3):223–253

Gonzalez-Velárde J, Laguna M (2002) Tabu search with simple ejection chains for coloring graphs. Ann Oper Res 117(1–4):165–174

Huang W, Ye T (2011) Global optimization method for finding dense packings of equal circles in a circle. Eur J Oper Res 210(3):474–481

Jaumard B, Meyer C, Thiongane B (2006) ILP formulations for the RWA problem for symmetrical systems. In: Handbook for optimization in telecommunications. Springer, New York, pp 637–677

Jaumard B, Meyer C, Thiongane B (2007) Comparison of ILP formulations for the RWA problem. Opt Switch Netw 4(3):157–172

Jaumard B, Meyer C, Thiongane B (2009) On column generation formulations for the RWA problem. Discret Appl Math 157(6):1291–1308

Krishnaswamy RM, Sivarajan KN (2001) Algorithms for routing and wavelength assignment based on solutions of lp-relaxations. IEEE Commun Lett 5(10):435–437

Lee K, Kang KC, Lee T, Park S (2002) An optimization approach to routing and wavelength assignment in wdm all-optical mesh networks without wavelength conversion. ETRI J 24(2):131–141

Lü Z, Huang W (2009) Iterated tabu search for identifying community structure in complex networks. Phys Rev E 80:026130

Manohar P, Manjunath D, Shevgaonkar R (2002) Routing and wavelength assignment in optical networks from edge disjoint path algorithms. IEEE Commun Lett 6(5):211–213

Martins AX, Duhamel C, Mahey P, Saldanha RR, de Souza MC (2012) Variable neighborhood descent with iterated local search for routing and wavelength assignment. Comput Oper Res 39(9):2133–2141

Noronha TF, Ribeiro CC (2006) Routing and wavelength assignment by partition colouring. Eur J Oper Rese 171(3):797–810

Noronha TF, Resende MG, Ribeiro CC (2008) Efficient implementations of heuristics for routing and wavelength assignment. In: Proceedings of 7th International Workshop on Experimental Algorithms (WEA 2008), C.C. McGeoch (Ed.), LNCS, vol. 5038, pp. 169–180

Noronha TF, Resende MG, Ribeiro CC (2011) A biased random-key genetic algorithm for routing and wavelength assignment. J Global Optim 50(3):503–518

Ramaswami R, Sivarajan KN (1995) Routing and wavelength assignment in all-optical networks. IEEE/ACM Trans Netw (TON) 3(5):489–500

Skorin-kapov N (2007) Routing and wavelength assignment in optical networks using bin packing based algorithms. Eur J Oper Res 177:1167–1179

Wang L, Huang W (2005) A new local search algorithm for job shop scheduling problem. Chin J Comput 28(5):60–67

Wang H, Huang W, Zhang Q, Xu D (2002) An improved algorithm for the packing of unequal circles within a larger containing circle. Eur J Oper Res 141(2):440–453

Wu Q, Hao JK, Glover F (2012) Multi-neighborhood tabu search for the maximum weight clique problem. Ann Oper Res 196(1):611–634

Ye T, Xu R, Huang W (2011) Global optimization of binary lennard-jones clusters using three perturbation operators. J Chem Inf Model 51(3):572–577

Zang H, Jue JP, Mukherjee B et al (2000) A review of routing and wavelength assignment approaches for wavelength-routed optical wdm networks. Opt Netw Mag 1(1):47–60