Load Balancing in WDM Networks through Adaptive Routing Table Changes

Mauro Brunato, Roberto Battiti, and Elio Salvadori

Università di Trento, Dipartimento di Matematica

via Sommarive 14, I-38050 Pantè di Povo (TN), Italy

battiti|brunato|salvador@science.unitn.it

Abstract. In this paper we develop Load Balancing algorithms for IP-based Optical Networks. The considered networks are based on a routing protocol where the next hop at a given node depends only on the destination of the communication. Our algorithm (RSNE - Reverse Subtree Neighborhood Exploration) performs at each iteration a basic change in the network by reconfiguring only a single entry in the routing table of a single node. We study the performance of our algorithm in realistic networks under static and dynamic traffic scenarios. Simulation results show a rapid reduction of the congestion for static networks. The performance of the incremental scheme while tracking a changing traffic matrix is comparable to that obtainable through the complete re-optimization of the traffic.

Keywords: WDM, load balancing, local search, adaptive routing

Corresponding author:

Mauro Brunato

Università di Trento, Dipartimento di Matematica

via Sommarive 14, I-38050 Pantè di Povo (TN) — ITALY

Phone: +39 0461 88 2039 Cellular: +39 349 6085459

Fax: +39 0461 88 1624 Email: brunato@science.unitn.it

1 Introduction

Wavelength Division Multiplexing (WDM) and Generalized Multi Protocol Label Switching (G-MPLS) have been proposed to support the growing bandwidth demand caused by the exponential Internet growth and to permit suitable traffic engineering. In WDM networks, a wavelength is assigned to each connection in such a way that all traffic is handled in the optical domain, without any electrical processing on transmission [3].

Current advances in optical communication technology are rapidly leading to flexible, highly configurable optical networks. The near future will see a migration from the current static wavelength-based control and operation to more dynamic IP-oriented routing and resource management schemes. Future optical networks designs should probably be based on fast circuit switching, in which end-to-end optical pipes are dynamically created and torn down by means of signaling protocols and fast resource allocation algorithms.

IP modifications are being proposed to take QoS requirements into account and to integrate the IP protocol within the optical layer. At the same time, a generalized version of Multiprotocol Label Switching (G-MPLS) is being developed to enable fast switching of various type of connections, including IP based lightpaths. As soon as protocol modifications can ensure different QoS levels at the IP level, more and more statically allocated traffic can be transmitted on the dynamic portion of the network leading to an all optical and fully dynamic G-MPLS controlled optical cloud [20]. In this scenario it is necessary to study the impact of routing mechanisms typical of the IP world, by analyzing the possible integration of label switching techniques (MPLS) with the current optical switching architectures. In addition, it is impor-

tant to study criteria and algorithms to decide when and how lightpath allocation and release requests are generated in the presence of data traffic.

The optimization in the usage of scarce resources in optical networks with WDM technology leads to the problems of packet routing (in packet networks) and of creating virtual connections by considering both routing and wavelength assignment. Some seminal papers are [5], [18], [16], [1], [23]. A review of algorithms for designing virtual topologies directly on the optical layer is presented in [9]. The network evolution in terms of traffic amount and flexibility requirements indicates that mesh networks must be considered instead of typical SDH/SONET-like ring topologies, thus opening a more complex scenario. A novel heuristic approach for the design of WDM networks under static traffic demands is described in [12]. The work [2] considers the physical topology as completely assigned and exploits the resources of the optical layer for the design of the virtual topology. Heuristics for the design of virtual topologies, based on greedy principles [17] or on linear programming [7] have been recently discussed.

Routing that takes into account the combined topology and resource usage information at the IP and optical layers [6], with constraints on the maximum delay or number of hops, is an area that deserves additional exploration. [22] investigates distributed control mechanisms for establishing all-optical connections in a wavelength routed WDM network: an approach based on link-state routing, and one based on distance-vector routing. A novel algorithm for integrated dynamic routing of bandwidth guaranteed paths in MPλS networks is developed in [6]. Another fundamental issue when designing network algorithms is their ability to function with the limitations of a distributed environment, i.e. local and delayed information; work [10]

considers the case in which nodes explore a limited portion of the search space by considering only a certain number of links along a path.

In the following sections, first we summarize the technological context of our proposal in Section 2, then we introduce the Reverse Subtree Neighborhood Exploration (RSNE) algorithm in Section 3 and discuss some variations and possible implementations in Section 4. Finally Section 5 analyzes simulation results, considering both the static and the dynamic traffic cases.

2 Adaptive routing and load balancing

The purpose of Load Balancing in WDM based networks is to reduce the congestion in the network. The congestion is related to delays in packet switching networks, and therefore reducing congestion implies better quality of service guarantees. In networks based on circuit switching (see for example the G-MLPS protocol), reducing congestion implies that a certain number of spare wavelengths are available on every link to accommodate future connection requests or to maintain the capability to react to faults in restoration schemes. In addition, reducing congestion means reducing the maximum traffic load on the electronic routers connected to the fibers.

Load balancing in WDM networks consists of two subproblems: the lightpath connectivity and the traffic routing problem. The routing problem has its origin at the beginning of the networking research, see [15] for a review of previous approaches to the problem. In particular, adaptive routing, that incorporates network state information into the routing decision is considered in [14] in the context of all-optical networks, while previous work on state-dependent routing with trunk reservation in traditional telecommunications networks is considered in [13]. It is also known that flow deviation methods [4], although computationally demanding, can be

used to find the optimal routing that minimizes the maximum link load for a given network topology.

Because global changes of the logical topology and/or routing scheme can be disruptive to the network, we consider algorithms that are based on a sequence of small steps (i.e., on local search from a given configuration). In [8] "branch exchange" sequences are considered in order to reach an optimal logical configuration in small steps, upper and lower bounds for minimum congestion routing are studied in [21] that also proposes variable depth local search and simulated annealing strategies. Strategies based on small changes at regular intervals are proposed in [15].

Our technological context is that of dynamic lightpath establishment in wavelength-routed networks reviewed in [22]. We therefore assume a mechanism to assign resources to connection requests, that must be able to select routes, assign wavelengths and configure the appropriate logical switches, see also [6] for integrated IP and wavelength routing and [11] for a blocking analysis in the context of *destination initiated reservation*.

This paper describes a preliminary investigation on protocols that consider IP-like routing strategies, where the next hop at a given node is decided only by the destination of the communication. In particular, we consider a basic change in the network that affects a single entry in a single routing table. In the context of all-optical networks this is relevant for optical packet switching networks, or for circuit switching networks (e.g. based on G-MPLS) where the optical cross-connects allow arbitrary wavelength conversion.

The focus is to study basic mechanisms in a simplified context. We plan to extend the work in the future by considering more general routing mechanisms (label switched paths in G-MPLS)

and limited or no wavelength conversion. In addition we plan to analyze the blocking probability when the traffic requests exceed the network capability.

By *physical topology* we mean the actual network constituted by passive or configurable optical nodes and their fiber connections. The lightpaths between the electronic routers, determined by the configuration of the OADM's and transmitters and receivers on each node, constitute the *logical topology*. The *traffic pattern* is available as an $N \times N$ matrix (N being the number of nodes in the network) $T = (t_{ij})$ where t_{ij} denotes the number of lightpaths (or the number of traffic load units) required from node i to node j. We assume that the entries t_{ij} are non-negative integers and $t_{ij} = 0$ if $i = j$. A *routing table* is an array, associated to each node of the network, containing next-hop information required for routing. In the following we shall consider IP-like routing, where the next hop is maintained for each possible destination, regardless of the index of the source node. For a given traffic pattern and routing tables associated to the nodes, the sum of the number of lightpaths passing through each link is called the *virtual load* of the link. Finally, the maximum virtual load on each link of a path is called the *congestion* of this path. The maximum virtual load on each link of a network is called the *congestion* of the network.

The Load Balancing problem can be defined as follows.

LOAD BALANCING — Given a physical network with the link costs and the traffic requirements between every pair of source-destination (number of lightpaths required), find a routing of the lightpaths for the network with the least congestion.

3 Local Search for the Load Balancing Problem

The basic idea of the new Load Balancing scheme is as follows: we start by finding the shortest path routing and try to minimize the congestion of the network by trying some local moves. For each tentative move we re-route part of the load on the most congested link in the network and accept that local move only if it reduces the load on the network. We iteratively find the most congested edges and re-route the traffic passing through these edges, until no more re-routing is possible within network constraints (every re-routing attempt ends up with either a disconnected network or a loop), or a fixed maximum number of iterations is reached.

Before explaining and discussing the heuristic algorithm in detail, we give some definitions and explanations of the parameters in use. We maintain the set *candidatePathSet* containing all paths that are candidate to replace those passing through the most congested links of the network. It consists of all paths having enough space to route the virtual load removed from the most congested link, without causing their virtual load to be higher than the current congestion of the network. This set is emptied at each iteration of the algorithm. Given all routing tables, every node d identifies a spanning tree of the network, namely the tree composed of all links that carry lightpaths addressed to d (one has a tree because of the destination based routing). We are interested in identifying a subtree of this tree, and we use the function *routingTree*(d,r) returning the subtree rooted in node r of the routing tree having destination node d. It is the grey-shaded tree in Fig. 2, and it contains all and only nodes whose communications directed to destination d pass through node r. The function *shortestPathRouting*(*network*) calculates the shortest path tree for each destination node and returns the corresponding routing table as a

matrix. *rTable*[*n*] is the routing table of node *n*, whose *i*-th entry *rTable*[*n*][*i*] is the next-hop node index for lightpaths passing through node *n* and with destination *i*. Finally, function *calculateLoad*(*network*,*traffic*,*rTable*) returns the network congestion given the network topology, the traffic pattern and the current routing scheme. The function also returns the set of links having maximum loads.

We show in Fig. 1 an outline of our Local Search algorithm used for the Load Balancing problem; in the rest of the paper we shall refer to it as *Reverse Subtree Neighborhood Exploration* (`RSNE`).

The initialization section (lines 1–2) starts by generating the routing tables through the application of the Shortest Path Routing algorithm to the specific network. In this particular case, the costs of all the edges have been considered as uniform. Using the function *calculateLoad* we initially calculate the load on each link of the network, the initial value of *congestion* (from which the local search algorithm starts its research of the minimum) and the set of congested links *congestedLinkSet*. Then *candidatePathSet* is empty at the beginning of each iteration.

The local search algorithm (lines 3–23) consists of two distinct parts. First, a set of alternative paths where to re-route part of the traffic passing through the most congested links is found (lines 6–17): then we correct the routing tables of the network (lines 18–22) and start over again by considering the new congested links.

The first part includes the core of `RSNE` algorithm. Refer to Figure 2 for a visual reference. We consider each congested link in *congestedLinkSet* (loop at lines 6-17); let us identify this link with its endpoints (*cFrom*,*cTo*). Then we must iterate through all lightpaths using that link. We use two nested loops: the first (line 7) scans the routing table of node *cFrom* looking for all

destination nodes *dest* using that link; the second (line 8) scans all nodes *src* whose lightpaths directed to *dest* run through *cFrom*. These nodes identify the subtree rooted in *cFrom* of the routing tree having destination *dest*.

For every (*src*,*dest*) pair whose lightpath goes through the link (*cFrom*,*cTo*), we try to reroute such lightpath by altering the routing table in *src*. To do this, we temporarily remove the load from the current route (line 9) and iterate through all downstream neighbors *nb* of *src* calculating the maximum load that would be caused by rerouting the lightpath, provided that the new route does not end up in a cycle and that the congested edge is avoided. The best alternate paths, in terms of maximum load, are collected into the candidate set *candidatePathSet*. In particular, the current minimum is stored in *bestCandidateLoad*. If the load obtained after this traffic re-route is equal to *bestCandidateLoad*, then the re-route is added to the candidate set (lines 12-13); if it is smaller, the candidate set is re-initialized to the current re-route and its load is stored as the new best (lines 14-16). At the end of the alternate paths research, the partial load associated to the path originating in *src* and terminating in *dest* is reallocated (line 17), in order to allow the search of new paths with different source nodes *src* (line 8).

In the second part of the `RSNE` algorithm, if the resulting set *candidatePathSet* is not empty then one random solution is selected from it (line 19), and the routing table of the network is updated (line 20). Finally, a new value of *congestion* and the relative set of most loaded links *congestedLinkSet* is calculated again in order to start a new search of alternate paths through the network.

Note that the local search algorithm continues looking for better values of *congestion* until the set of candidate re-routes *candidatePathSet* is empty (line 22), or until a given number of iterations has been performed (line 23).

A more simplified version of the algorithm, which we shall call *Restricted Neighborhood Exploration* (`RNE`) can be implemented by considering only node *cFrom* as candidate for routing table modifications (we eliminate the grey part of Fig. 2). This corresponds to the elimination of the loop structure on line 8, which scans the *cFrom*-rooted subtree, setting *src* equal to *cFrom*. The rationale for `RNE` is to avoid a large tree exploration and to keep modifications as near as possible to the congested link. In fact, while rerouting at *cFrom* removes a whole bundle of lightpaths from the link, doing the same at some upstream node causes a smaller reduction of the load. Moreover, a distributed version from `RNE` would be simpler (see Sec. 4.2). On the other hand, simulations in Sec. 5 show that, unless very few iterations are allowed before halting, performance of `RNE` is significantly worse than `RSNE`.

4 Incremental and distributed implementations

4.1 Dynamically evolving traffic

Local search heuristics can be seen as stepwise refinements of an initial solution by slight modifications of the system configuration. In our case, the `RSNE` algorithm starts from a shortest path routing scheme and changes at every step a routing table entry of a single node in the matrix. By performing many such changes, the system stabilizes to a low congestion configuration.

This iterative scheme is appropriate for a dynamic environment where traffic requirements evolve with time. In particular, if changes in the traffic matrix are reasonably smooth[1] even a small number of steps of the RSNE algorithm in Fig. 1 are sufficient to keep the system in a suitable state as the traffic matrix changes. Of course, only lines 2-23 must be executed, because we don't want to restart from scratch by calculating the shortest path routing tables. Moreover, a very low number of iterations of the outer loop (lines 3-23) must be performed at each step, i.e. MAXITER must be very small to avoid excessive traffic disruption. In the following, we refer to the incremental algorithm as *Incremental* RSNE with k iterations per step: I−RSNE(k).

Simulations discussed in Section 5 show that even a single iteration of the algorithm yields good results under a fairly generic traffic model. The number of iterations of the algorithm is equivalent to the number of routing table entry modifications in the systems; thus, a very limited number of routing table entries must be modified as traffic evolves in order to keep congestion at low levels.

A similar approach has been proposed in [19], where branch-exchange methods are proposed for a local search heuristic; however, the type of local modification is quite different from our proposal.

4.2 Distributing the algorithm

The RSNE and I−RSNE(k) algorithms described in Fig. 1 and in the previous section are centralized schemes: good candidates are searched throughout the system by scanning all most

[1] The assumption is reasonable even though IP traffic is known to be bursty: in fact, traffic requirements are given as an average over a certain amount of time, with some marginal capacity left to accommodate traffic peaks.

loaded links and all lightpaths passing through them. However, the choice of the link whose load should be decreased can be implemented as a distributed scheme, while all other candidate selection loops (exploring all possible destinations, exploring the source subtree, finding an alternate route through neighbors) can be implemented by simple token-passing mechanisms or by assigning different time slots to nodes in order to avoid concurrent execution of critical parts of the code.

Distributed schemes usually suffer from high message complexity, but simple randomization techniques can help to reduce the amount of messages traveling through the system.

We can avoid a global message exchange session among nodes by implementing some link-state periodic messaging technique, so that every node maintains a view of the global status of the system. In general, this knowledge does not always reflect the current system status, because every node updates its information about different parts of the network at different times. Partial knowledge, however, is enough to set a threshold so that a node initiates the load reduction protocol when the load on a downstream edge exceeds it. Of course, two nodes may start at the same time; in this case, either the domain of their protocols are disjoint, and they can complete their load reduction algorithm without interference, or some nodes will detect the interference and act appropriately. Even the load reduction scheme itself can be randomized, for instance only a subset of all possible lightpaths can be examined for rerouting, in order to greatly reduce the message complexity of the system.

We are currently studying and experimenting various distributed and randomized versions of the I−RSNE(k) algorithm that will be presented in an extended version of this work.

5 Simulation results

5.1 Static tests

To test the algorithm we performed two sets of tests, static and dynamic. The first, using a static traffic matrix, allowed us to explore the convergence speed of the `RSNE` and `RNE` algorithms, showing that the first achieves lower congestion values.

Figure 3 plots the best congestion value against the number of steps for one run of the `RNE` and `RSNE` algorithms; here the 24-node regional network considered in [24] was used, together with the 1995 traffic pattern presented in the same work. It turns out that the more complete `RSNE` algorithm outperforms its simpler version. `RNE` provides a better result in the initial phase, probably because the algorithm is forced to move larger portions of load from edge to edge, achieving temporary better results but ending up with a complex routing scheme that cannot be improved.

The above simulation gave a maximum hop length equal to 7 (i.e. a lightpath needs to travel 7 links from source to destination) at each iteration. This is the minimum, because it also results from the shortest path routing assignment that initiates both algorithms.

5.2 Dynamic traffic

Next we considered another model, a 14-nodes NSF network with traffic evolving in time. To generate dynamic traffic we followed a model similar to that described in [15]. Given two positive integers N and Δ, we consider a sequence of $N\Delta+1$ traffic matrices $(T^0, T^1, \ldots, T^{NDelta})$ where matrices $T^{k\Delta}$, $k = 0, 1, \ldots, N$ are random and independently generated, forming an i.i.d.

subsequence: every entry of these matrices is a random integer between 0 and 5. All other matrices are rounded linear interpolations of the immediately adjacent random matrices; in other words, given $h = 0, \ldots, \Delta - 1$ and $k = 0, \ldots, N-1$, entry $T_{ij}^{k\Delta+h}$ of matrix $T^{k\Delta+j}$ is computed as follows:

$$T_{ij}^{k\Delta+h} = \text{round}\left[\left(1 - \frac{h}{k}\right) T_{ij}^{k\Delta} + \frac{h}{k} T_{ij}^{(k+1)\Delta}\right].$$

We ran a series of 10 tests on a sequence of traffic matrices modeling the dynamic traffic pattern described above; the composition of each test was the following:

- a run of the `RSNE` algorithm, where the algorithm was reapplied from scratch to every matrix of the sequence;
- a run of `I-RSNE`(1) where the initial routing scheme was obtained from a complete run of the `RSNE` algorithm, while in all subsequent steps a single iteration of the incremental algorithm was performed;
- a run of the `I-RSNE`(3) algorithm, as above, but with three iterations of the incremental algorithm per step.

All runs were executed over the same traffic pattern. The average results are shown in Fig. 4. As expected, the overall best results are achieved by using the complete algorithm, whereas the incremental implementations suffer from a slightly higher congestion. However, the maximum difference between the implementations is about $6 - 7\%$, and occasionally the incremental technique outperforms the static one. Note also that increasing the number of iterations per step from 1 to 3 improves the result only marginally. This means that modifying a single entry in the

routing table of a single node at each step is enough to adequately follow the traffic pattern, at least with this traffic model.

6 Conclusions

We consider a simple Load Balancing algorithm for IP based Optical Networks, based on Local Search where a basic move modifies a single entry in the routing table of a node.

So far, our work has been restricted to a simplified context; however, many extensions can be envisioned, in particular when considering specific properties of the optical medium. For instance, a lightpath must be reconverted at every node in order to have its address examined, and to determine its next hop in an IP-like fashion (Optical Packet Switching). The problem of networks having links of different capacities will also be considered in future extensions.

The `RSNE` algorithm explores all possible improvements before taking a step. Further investigation will determine how the quality of the solutions deteriorates if a randomized approach such as the one discussed in Section 4 is followed in order to distribute the algorithm.

Acknowledgments

We would like to thank Imrich Chlamtac and Jason Jue of the University of Texas at Dallas, for their interesting and fruitful discussions with the authors about the subject of this work.

References

1. D. Banerjee and B. Mukherjee. A practical approach for routing and wavelength assignment in large wavelength-routed optical networks. *IEEE Journal on Selected Areas in Communications*, 14(5):903–908, 1995.

2. S. Bregni, U. Janigro, and A. Pattavina. Optimal allocation of limited optical-layer-resources in wdm networks under static traffic demand. In *Proceedings of Globecom 2001*, pages 25–29, San Antonio, Texas, 2001.

3. I. Chlamtac, A. Ganz, and G. Karmi. Lightpath communications: A novel approach to high bandwidth optical wan-s. *IEEE Transactions on Communications*, 40(7):1171–1182, 1992.

4. L. Fratta, M. Gerla, and L. Kleinrock. The flow deviation method: An approach to store-and-forward communication network design, 1973.

5. A. Ganz and X. Wang. Efficient algorithm for virtual topology design in multihop lightwave networks. *IEEE/ACM Transactions on Networking*, 2(3):217–225, 1994.

6. M. Kodialam and T. V. Lakshman. Integrated dynamic IP and wavelength routing in IP over WDM networks. In *Proceedings of IEEE Infocom 2001*, pages 358–366, 2001.

7. R. M. Krishnaswami and K. N. Sivarajan. Design of logical topologies: A linear formulation for wavelength routers with no wavelength changer. *IEEE/ACM Transactions on Networking*, 9(2):186–198, 2001.

8. J. Labourdette and A. Acampora. Logically rearrangeable multihop lightwave networks. *IEEE Trans. on Commun.*, 39:1223–1230, Aug 1991.

9. E. Leonardi, M. Mellia, and M. Ajmone Marsan. Algorithms for the topology design in wdm all-optical networks. *Optical Networks Magazine "Premiere Issue"*, 1(1):35–46, Jan 2000.

10. Ling Li and Arun K. Somani. Dynamic wavelength routing using congestion and neighborhood information. *IEEE/ACM Transactions on Networking*, 7(5):779–786, 1999.

11. K. Lu, G. Xiao, and I. Chlamtac. Blocking analysis of dynamic lightpath establishment in wavelength-routed networks, 2001, submitted to ICC2002.

12. G. Maier, A. Pattavina, L. Roberti, and T. Chich. Static-lightpath design by heuristic methods in multifiber wdm networks. In *Proceedings of OPTICOMM 2000*, pages 64–75, Dallas, TX, 2000.

13. D. Mitra, R. Gibbens, and B. Huang. State-dependent routing on symmetric loss netowkrs with trunk reservations. *IEEE Transactions on Communications*, 41(2):400–411, 1993.

14. Ahmed Mokhtar and Murat Azizoğlu. Adaptive wavelength routing in all-optical networks. *IEEE/ACM Transactions on Networking*, 6(2):197–206, 1998.

15. Aradhana Narula-Tam and Eytan Modiano. Dynamic load balancing in wdm packet networks with and without wavelength constraints. *IEEE Journal of Selected Areas in Communications*, 18(10):1972–1979, Oct 2000.

16. R. Ramaswami and K. N. Sivarajan. Design of logical topologies for wavelength-routed optical networks. In *Proceedings of INFOCOM 1995*, 1995.

17. A. Schupke and D. Sellier. Lightpath configuration of transparent and static WDM networks for IP traffic. In *Proceedings of ICC2001*, 2001.

18. Skorin-Kapov and J.-F. Labourdette. On minimum congestion routing in rearrangeable multihop lightwave networks. *Journal of Heuristics*, 1:129–145, 1995.

19. Jadranka Skorin-Kapov and Jean-François Labourdette. On minimum congestion routing in rearrangeable multihop lightwave networks. *Journal of Heuristics*, 1:129–145, 1995.

20. C. Xin, Y. Ye, T.S. Wang, and S. Dixit. On an IP-centric control plane. *IEEE Communications Magazine*, 39(9):88–93, 2001.

21. Bülent Yener and Terrance E. Boult. A study of upper and lower bounds for minimum congestion routing in lightwave networks. In *Proceedings of IEEE INFOCOM*, pages 138–149, 1994.

22. H. Zang, J.P. Jue, L. Sahasrabuddhe, R. Ramamurthy, and B. Mukherjee. Dynamic lightpath establishment in wavelength routed networks. *IEEE Communications Magazine*, 39(9):100–108, 2001.

23. Z. Zhang and A. S. Acampora. A heuristic wavelength assignment algorithm for multihop wdm networks with wavelength routing and wavelength re-use. *IEEE/ACM Transactions on Networking*, 3(3):281–288, 1995.

24. Zhensheng Zhang and Anthony S. Acampora. A heuristic wavelength assignment algorithm for multihop WDM networks with wavelength routing and wavelength re-use. *IEEE/ACM Transactions on Networking*, 3(3):281–288, June 1995.

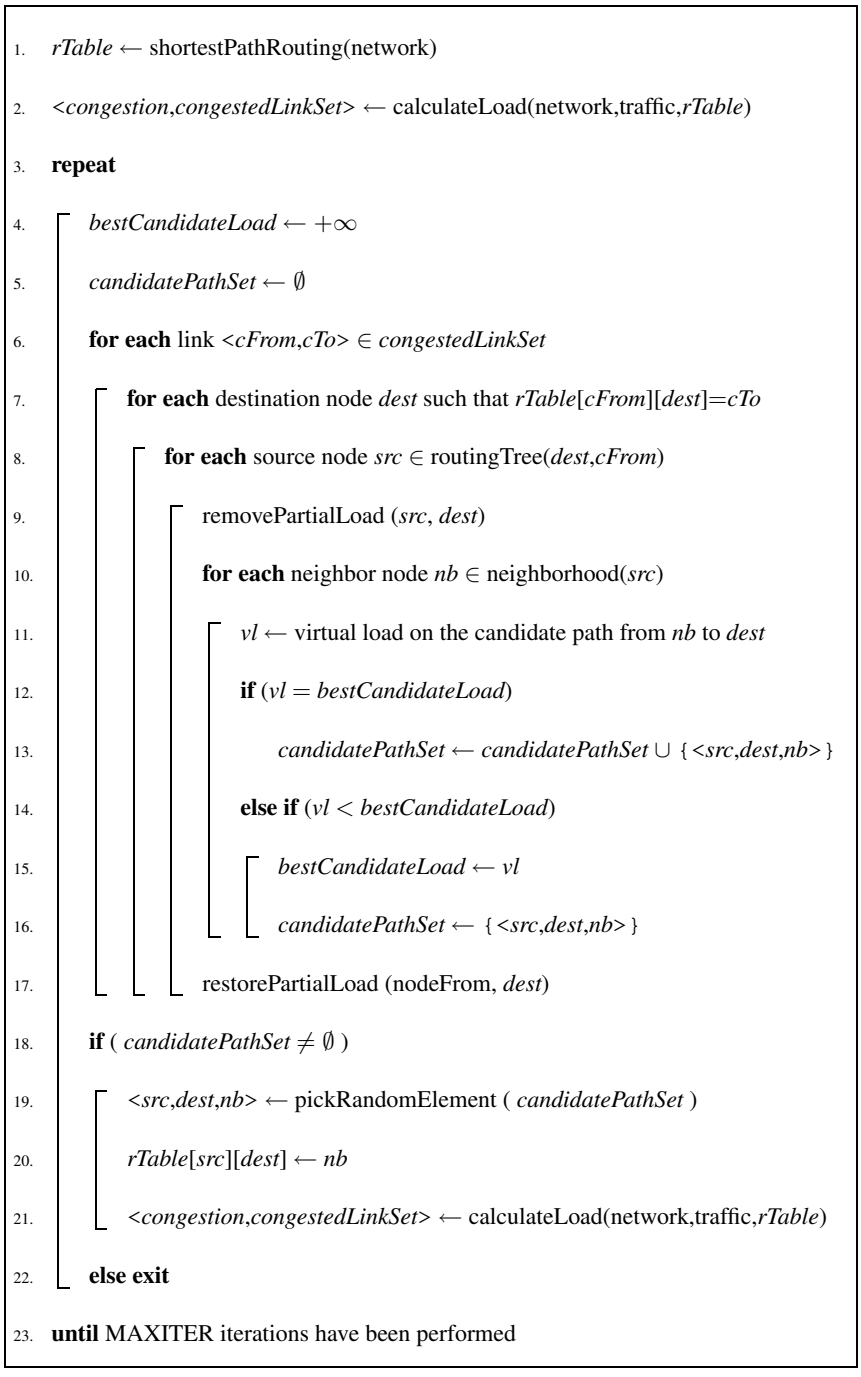

Fig. 1. the Local Search RSNE algorithm

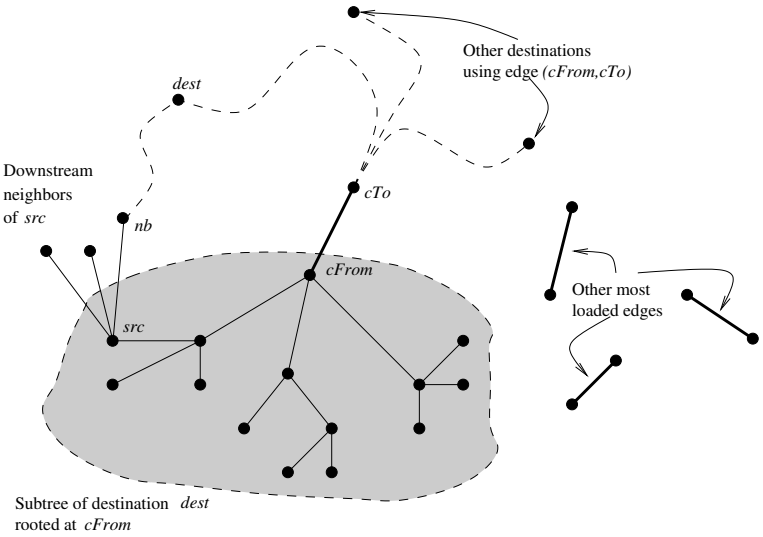

Fig. 2. The search space for a move of the RSNE algorithm

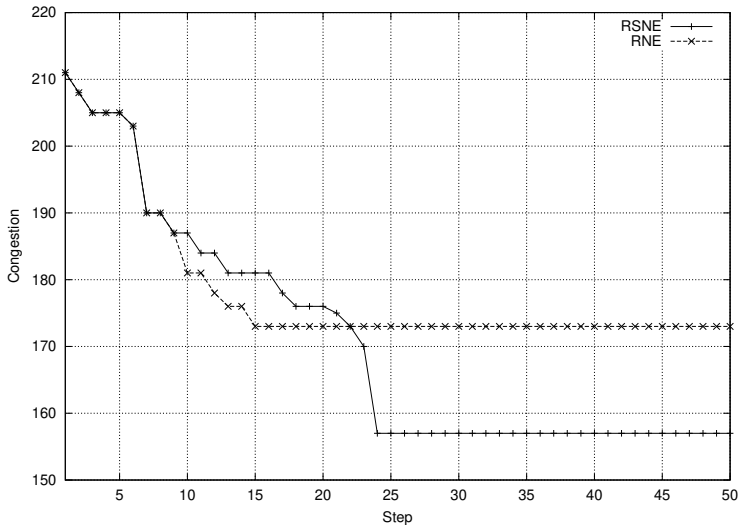

Fig. 3. Comparison between RSNE and RNE algorithms.

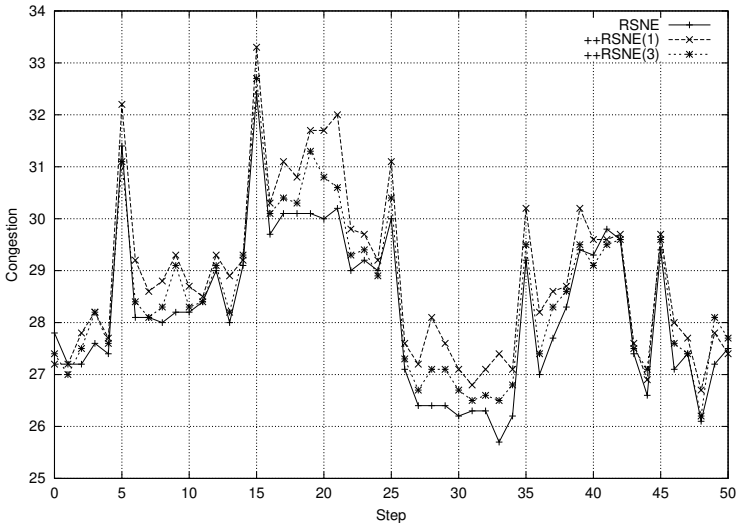

Fig. 4. Comparison among RSNE and I−RSNE(k).