Shortest Path Green Routing and the Importance of Traffic Matrix Knowledge

Mariusz Kamola
NASK - Research and Academic Computer Network
Warsaw, Poland
Email: Mariusz.Kamola@nask.pl
Institute of Control and Computation Engineering
Warsaw University of Technology
Warsaw, Poland
Email: M.Kamola@ia.pw.edu.pl

Piotr Arabas
NASK - Research and Academic Computer Network
Warsaw, Poland
Email: Piotr.Arabas@nask.pl
Institute of Control and Computation Engineering
Warsaw University of Technology
Warsaw, Poland
Email: P.Arabas@ia.pw.edu.pl

***Abstract*—Energy-efficient traffic engineering for wired network poses a number of difficulties due to equipment and protocols diversity and complexity. If emerging optimization are to be formulated exactly, they are intractable. This is why numerous heuristics for approximate solutions are proposed. They differ in assumptions on the routing scheme in the network, and also on the availability (and usefulness) of traffic-related or measurement-based power consumption data. Those most interesting w.r.t. authors' work are presented and classified. The authors propose a number of strategies for switching the links on or off, which are still worthy consideration — while the routing scheme stays in control of some shortest-distance mechanism, like OSPF. Operation of the algorithms is verified for two standard test cases, resulting in proposed further improvements and hints on implementation details.**

***Keywords*—OSPF, energy-efficient routing, heuristic algorithms, traffic matrix**

I. Introduction

Demand for energy consumption saving in wireless as well as wired data networks is indisputable, although motivations and means of performing one differ in both environments. Saving battery power has become the necessity for portable devices and pushed green networking research forward, while wired power saving technologies lagged, lacking adequate motivation. Most of present wireless applications where full-fledged green routing is really needed are sensor networks; technologies developed there [1], [2] have become the inspiration for solutions dedicated to wired networks. And vice versa, the results presented in this paper can be used for further development of routing strategies in sensor networks. However, with rapidly increasing data volumes consumed by mobile terminals it is obvious that massive wireless communication will be maintained only in access network — just because wired networking provides much more ways of saving energy.

The most obvious one (and impossible for wireless communication) is changing the medium — and it gives most impressive savings as well. However, no matter how green fiber optic links are, the packets must be processed at network nodes, and such processing can still be costly, depending on technology to some extent. Therefore, appropriate traffic engineering, with the goal to use optimal network configuration to handle the traffic and fulfill QoS guarantees is the next thing done by ISP after hardware upgrades.

Necessity to save energy by all means is articulated best in [3], where estimated savings are provided for each network type (core, metro and access). To complete the view, [4] gives an interesting and comprehensive survey of research on green networking technologies; also [5] presents important advancements in the field. Let us focus in the rest of the paper on green routing strategies alone. As stated in [4], power consumption of any elementary networking component (e.g. an interface) is a function of its usage (e.g. traffic load). In the extreme cases, the device can be energy-saving agnostic (power does not depend on load) or it can save energy perfectly (proportionally to traffic load). In reality, due to hardware and software specifics, the power consumption is stepwise and located between the two extremes. To make the matters computationally worse, complex networking appliances combine those component stepwise characteristics, rendering the problem of optimal network configuration an NP-hard one.

The practical approach to handle such complexity is to use some relaxation or other form of heuristic. For example, [6] applies branch and bound approach while performing traffic engineering of MPLS paths that match equipment energy functions optimally. A link component with stepwise energy profile is treated as a number of agnostic "sub-components". The resulting integer problem is iteratively relaxed to the continuous linear one, only to qualify gradually on/off state of each "sub-component" by a heuristics of some sort. Also, [7] utilizes information about location of energy steps to propose heuristic approach where arbitrary flows are rerouted so that devices loads are just on the verge of entering subsequent, higher energy states. Please note that both these approaches require a single data flow to be routed along an arbitrary path. Moreover, the earlier one also postulates traffic matrix to be known. One may argue that both requirements are quite strong, especially w.r.t. networking practice. Firstly, because shortest routes are calculated routinely by OSPF, which means that confluent paths will never diverge again. Secondly, research shows that accurate traffic matrix estimation is rather impossible due to interdependence of flows. This is why we propose to review and contribute to research on green shortest-path routing strategies, hopefully not requiring knowledge of traffic

978-1-4799-0756-4/13/$31.00

matrix.

The rest of the paper is organized as follows. Further in this section we present existing approaches to green routing that rely on shortest-path destination routing schemes. We compare their capabilities with do-nothing and "do-best" approaches, for typical power-saving scenarios. Then in Sec. II we present our proposition of green routing heuristics. Its operation in standard test cases gets described in Sec. III, where numerical results are provided. The conclusions are given in Sec. IV.

The algorithm proposed in [8] selects links or nodes to be switched off; then standard OSPF protocol with Dijkstra algorithm used for shortest path tree (SPT) computation does the rest of the job. The only question is which links should be downed so that OSPF gets as much savings as possible, without violation of QoS constraints. Two heuristics are analyzed: one is to detect cliques in network graph, and to reroute flows so that maximum number of links can be switched off. The other one is simply to put down a least used link. The simulation results show that the earlier approach yields much better performance. The algorithm has many practical advantages: in particular, it can be implemented in one router only, while the remaining ones in the area simply implement SPT calculated and broadcast by it.

GRiDA algorithm [9] utilizes agent-based approach, with each agent being run on and controlling a single router. Agents decide autonomously upon links (or even whole routers) that can be switched off in order to improve performance. Should powering down a link cause a congestion, the agent's wrong decision gets remembered for future, according to machine learning paradigm. Agent algorithm incorporates much practical experience and detailed knowledge of energy profile of the machine being under its control. Unlike in [8], it does not take into account information from the routing process itself, i.e. network topology and current routing paths.

An interesting alternative approach has been recently presented in [10], where one of routers assumes the coordinator role and executes a heuristic coordination algorithm, while other routers report their maximum link utilization, on which the coordination decisions are based.

Savings from link switching off combined with standard routing strategies differ, depending on the power saving capabilities of networking devices. Let us now consider a situation without link capacity constraints.

- *All links save power perfectly.* This case does not require any extra power saving algorithm. Shortest path routing with link metrics equal to cost of data unit transfer solves the problem optimally by choosing shortest paths — in the sense of sum of subsequent data transmission costs, in this case. Networks with energy-efficient link control algorithms (cf. eg. [11]) in off peak hours do not require any sort of extra energy saving logic.
- *All links are power agnostic.* In this another extreme case shortest path metric running freely will result in huge energy waste. For instance, take not so rare full-mesh topology: OSPF will activate *all* links, while the minimum number of spanning tree branches would be the number of nodes less one. The approach for optimal solution is to find the minimum (in the sense of total power consumed) spanning tree. Complexity of such algorithms [12], [13] is functional polynomial (FP). As for heuristic algorithms described in [8], [9], they will result in switching off as many links as to maintain connectivity. In case of *equal* energy consumption on all links, this will in fact result in a functional spanning tree, although it may take form of a daisy chain in the extreme case. In case where links power consumption differ, any local heuristic decision will be deceiving, depending on the scale of disproportion among link loads and link power consumptions.

With link capacity constraints considered, none of the above approaches can give optimum solution in neither case. This is because traffic has to be split to guarantee load balancing, and such functionality is just impossible with simple approaches like Dijkstra's. However, algorithms from [8], [9] are designed rather to save energy in off-peak hours, and not to guarantee maximum network throughput — the latter is usually achieved by appropriate network design.

II. Proposed framework and heuristics

We suggest to consider an architecture for green routing which is similar to those ones presented in the literature in the sense that links designated to be switched off or on are being selected by autonomous algorithm, and the resulting new traffic routes are being updated by standard routing schemes, like OSPF. Our approach is similar to [9] in that we propose to remember history of algorithm decisions; it is different in that our history is collective and much more limited one. We also do not plan to switch whole nodes, neither we use local energy profiles info. Our approach is similar to [8] in that we also take into account network topology and path information; it is different in that link state is our decision variable, instead of a path.

Let us denote nodes number by N, traffic on link l as y_l, link routing weight as w_l, and the binary routing matrix calculated by the routing process as $\boldsymbol{A}(\boldsymbol{w})$. The bold symbols we use denote matrices or vectors, with their elements written in ordinary font. Flows are sorted lexicographically to form a vector $\boldsymbol{x}$ so that traffic demand from node i to j is the element $x_{(i-1)N+j}$. We omit here graph topology data, relying further only on $\boldsymbol{A}$ resulting from it, and calculated by the routing process. We have $\boldsymbol{y} = \boldsymbol{A}\boldsymbol{x}$, i.e. $a_{l,k}$ is set to one if flow k traverses link l. By modifying $\boldsymbol{w}$ in subsequent algorithm steps we control the routing in the network; in particular, by setting $w_l = \infty$ we cause all flows to divert from link l and be routed otherwise.

The algorithm operates by constant supervision of the total network traffic, $\sum_l y_l$. Shall it change by a predefined fraction, e.g. due to seasonality, two possible kinds of reactions are envisaged:

- The total traffic has decreased. This means that an attempt should be made to switch off some more links.
- The traffic has increased. This means that some of the links probably need to be switched on.

For the decision to be made, the algorithm maintains three lists of links:

Y_{ON} links that must be permanently switched on in order to maintain network connectivity,

Y_{OFF} links that have been switched off in order to reduce power consumption,

Y_{BY} links that must be switched on in order to prevent congestion ("bypass" links).

In the event of increasing traffic the algorithm iteratively picks a link from Y_{OFF}, activates it and places it in Y_{BY} — until congestion is canceled. We consider here three ways of selecting the link to be switched on:

RL Roll back Last deactivated link: most recently deactivated link from Y_{OFF} gets active again. This method refers strictly to sequence of previous decisions; it requires Y_{OFF} to be really a list to pop the link index from. By doing so iteratively, it may arrive to the first link in Y_{OFF} that might be switched off a long time ago in situation different from the current one.

RB Congested link l is Relieved by activation of a link $m^\star$ that results in biggest decrease of routing paths number that traverse l — Blind to actual traffic matrix $\boldsymbol{x}$. Unlike RL, this method does not depend on actual sequence of previous decisions but rather on current state of the power saving mechanism determined by $Y_{\mathrm{ON}}, Y_{\mathrm{OFF}}, Y_{\mathrm{BY}}$. For each $m \in Y_{\mathrm{OFF}}$ it sets w_m to its original value and calculates a variant of routing $\boldsymbol{A}(\boldsymbol{w})$ *without implementing it* in the network, but only to find a bypass link $m^\star$ that minimizes $\sum_k a_{l,k}$. The link $m^\star$ can be perceived as most effective bypass for l w.r.t. the number of diverted flows.

RM Congested link l is Relieved by activation a link $m^\star$ that results in biggest decrease of traffic volume on l — by using traffic Matrix information $\boldsymbol{x}$. Analogously to RB, $m^\star = \arg\min \sum_k a_{l,k} x_k$. The link $m^\star$ can be perceived as most effective bypass for l w.r.t. the number of diverted traffic volume.

In the event of decreasing traffic the algorithm iteratively picks a link from Y_{BY} (or a link that does not belong to any of the above sets, in the initialization stage) and places it in Y_{OFF} — until congestion is observed. We consider four ways of selecting the link to be switched off:

LLL Least Loaded Link is switched off.

LTL Least Traversed Link, i.e. link with smallest number of shortest paths traversing it, is switched off. The decision to power down the link with smallest betweenness is rooted in social network analysis; intuitively, it reduces consequences of rerouting many flows and imposing much load on other active links. The number of shortest paths traversing each link is available directly from the routing process itself (e.g. OSPF).

LDB The link which will cause Least total Diversion of flows is switched off — Blind to actual traffic matrix $\boldsymbol{x}$. The total diversion is defined as increase of the total length of shortest paths w.r.t. the current total values. Numerically, the total length of paths is just the sum of $\boldsymbol{A}$'s elements. Analogously to RB and RM, this decision requires computation of $\boldsymbol{A}$ for as many variants of routing as there are possible links to be switched off.

LDM As for LDB, but with consideration to the actual traffic Matrix. In this case a link $m^\star$ is selected to be switched off, so that $m^\star = \arg\min ||\boldsymbol{A}\boldsymbol{x}||_1$, i.e. minimizing the resulting total traffic in the network.

Therefore, our proposed strategies cover a wide range of cases differing by available data: starting from the simplest one (LLL+RL), which is based on link load only, and which has been already considered in the former studies, going through LTL/LDB+RB, where routing information is consumed, and finally reaching LDM+RM, where traffic matrix knowledge is used, The last case was not considered in [8] and it was questioned as inefficient in [9]. Implementation of these strategies will naturally vary, but all of them control the operation of the existing routing scheme in the same way: by modifications of links weights. Each router is equipped with its controller that implements the algorithm and injects weights into the routing process. As the proposed strategies are deterministic and control links states sequentially, they will converge in natural way, by waiting for the next move (putting the selected link up or down), thus achieving identical $Y_{\mathrm{ON}}, Y_{\mathrm{OFF}}, Y_{\mathrm{BY}}$.

The algorithm input requirements vary. For LLL+RL, only link traffic measurements are required. Like in other papers, we propose them to be collected from interface statistics by relevant controllers, and distributed using e.g. OSPF opaque LSA user extensions. As this information is broadcast, it will not add hazards to controllers' decisions. In fact, the proposed algorithm is distributed in a way much similar to OSPF itself: all its local instances execute the same deterministic strategy, proceeding to the next step only if the previous one has been effectuated by the routing mechanism, whose output is under observation. Strategies based on the number of traversing paths will derive routing path and topology information straight from the local routing process. In cases where a number of routing variants have to be checked, these might be calculated e.g. by another "fake" routing process on the local router itself, or simulated inside the controller. Finally, strategies based on traffic matrix must estimate it first; the problem is not a trivial one because random sampling IP headers on interfaces deteriorates performance and is inaccurate. On the other hand, inferring $\boldsymbol{x}$ by using statistical methods can be spoiled by flows violating the assumptions (flows interdependence or unsupported distribution of an underlying stochastic process) — cf. [14]–[16].

III. Results

The proposed strategies have been tested in two example networks, whose models are available from [17], which is a repository widely used for benchmarking. We have chosen problem definitions for directed graphs, with defined node capacities and traffic matrix. Since no feasible solutions have been provided so far for the chosen problems due to their tight capacity constraints, those constraints are either not considered or relaxed by some factor in our experiments. We assume that, when in on state, power consumption is proportional to the

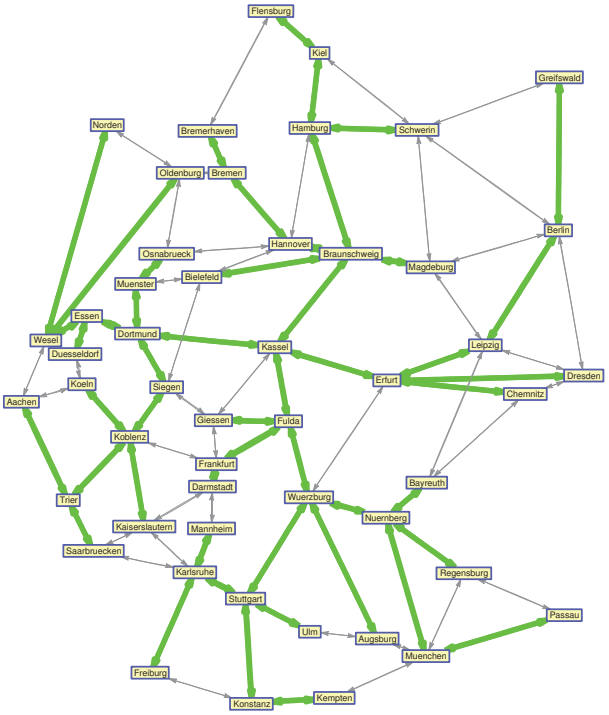


Fig. 1. Test network with exemplary spanning tree shown.

traffic transmitted, plus constant power due to link activation — and that all links are identical w.r.t. their energy efficiency. Being aware of such oversimplification, we also have to keep in mind that it helps in better understanding of the results. Unlike in other research, we decided not to define any exact energy values for links activation, neither for data unit transmission costs, thus looking at the problem as a bicriteria one: total network flow (costs) vs. number of links deactivated (savings).

In preliminary comparison, four methods of link deactivation were run against a SNDlib problem `germany50--D-B-L-N-C-A-N-N`. Capacity constraints were removed to allow methods switch links off until still active nodes form a spanning tree. As observed in Sec. I, all methods will cause switching off as many links as possible, 39 in this case. `LTL` has performed best in minimizing total network traffic; the resulting spanning tree is shown in Fig. 1 in green color.

The sum of all link loads in subsequent algorithm steps, for all four strategies of switching of links, is shown in Fig. 2. The sum of loads has the meaning of total network power consumption for links perfectly saving energy with equal data unit costs. For constant traffic matrix, all four strategies try deactivation of further links in each step; if the network graph connectivity is not lost, it results in increased $\sum_i \boldsymbol{y}_i$. Flat graph parts represent unsuccessful attempts to switch off links, which cause the graph being split into two parts. Naturally, in real network each successful step results also in savings from putting links to sleep. Such savings prevail in the initial part of the scenario, as they occur frequently and at the cost of almost negligible increase of the overall link load.

The actual number of steps is always more than 39 because

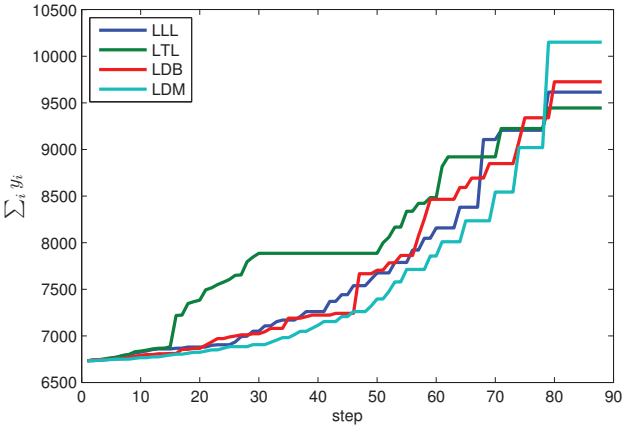
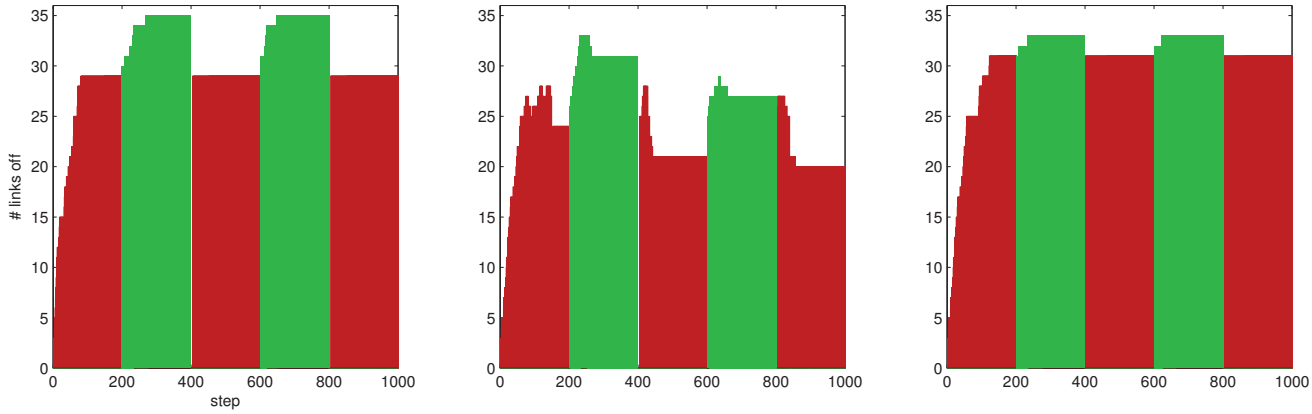


Fig. 2. Total network flow in subsequent steps of algorithms switching off the links.

trials of deactivating links that result in loss of network connectivity also count in. We can observe that, although `LTL` yields at last the smallest increase of the costs, it performs worst in the meantime. Remember that we did not count capacity constraints, and that they are usually hit quite early. It is therefore the medium part of the graph that should receive most attention: observe that until 50th step the links are deactivated steadily (except for `LTL` itself), with almost negligible increase of the transmission costs. (Those links are usually located on the network peripheries). With this in mind, we observe that the simplest strategy, `LLL` is almost on par with `LDB`, which is much more demanding technologically. The reason is that `LDB` counts all flows as equal, which is not our case. Therefore, `LDB` applies complex decision scheme based on wrong data, which must be misleading. With actual traffic matrix known and utilized, `LDM` performs best almost all the time.

The results shown so far suggest strategies `LTL` and `LDB` to be excluded from further investigations, the former for its inferior performance, the latter for its performance inadequate to algorithm complexity. Also `RB` will not be analyzed anymore, as the counterpart for `LDB`. We have therefore checked how `LLL+RL` and `LDM+RM` behave in presence of capacity constraints. Instead of adopting link capacities from original problem formulation (it still waits for optimal solution as regards source-destination routing), we have set capacities for all links to be equal 220 — i.e. slightly above the maximum link load observed for the original shortest-path routing, which was 216. The constraints redefinition was made to assure that the initial routing matrix found by OSPF would always be feasible. Such setting for capacity constraints corresponds to high load network conditions. Other condition, the off-peak hours, can be simulated by decreasing traffic matrix proportionally (or by "increasing" link capacities, which was, in fact, implemented in the experiments).

Results from numerical experiment of simulating alternating periods of high and low network utilization are shown in Fig. 3. The indicators of interest are the number of links switched off — in flat parts of the graph, where algorithm output stabilizes. The comparison of simple (Fig. 3(a)) and complex (Fig. 3(b)) control schemes is not in favor of the latter one — unlike for traffic costs in the unconstrained case.



(a) LLL+RL (b) LDM+RM (c) LDM+RL

Fig. 3. Number of links switched off in high load and low load periods (red and green colors, resp.) in "germany50" network.

Apparently, knowledge of the routing paths *and* the traffic matrix does not help. One of possible reasons is that RM uses a *set* of deactivated links, and *not a list*. All inactive links are treated as equal candidates to be switched on again — which turns out to be ineffective as sometimes an early deactivated link gets active, destroying the progress made. Moreover, performance of LDM+RM in subsequent phases deteriorates, thus demonstrating that algorithm switching the links off should rather retreat using the same route it got to the solution, in case of growing traffic.

It would be unreasonable to part with LDM, especially that it performed so well in unconstrained case. Instead, we tried a hybrid approach, LDM+RL. The simulation results are provided in Fig.3(c): now the algorithm is more stable. In off-peak periods it gives less savings than LLL+RL (do not forget the cost part, Fig. 2, where it performs better); in the peak period it switches substantially more links than LLL+RL. To confirm the results, an analogous series of experiments were performed for the model of GÉANT network, consisting of 22 nodes and 36 links. The graphs of the number of links switched off is shown in Fig. 4. Also here LDM combined with RL slightly overpasses both "pure" approaches; and it also retreats from wrong decisions more swiftly — cf. the initial peak around step 60: it took RM several steps to recover from overload conditions caused by switching off one of the links. Interestingly enough, the algorithm efficiency is not big in the first peak period, but it improves in the subsequent one.

IV. Conclusion

We have shown here that utilization of the traffic matrix and routing information to accomplish a complex heuristic routine for switching off links is not a straightforward task. Using just the routing info (LTL, LDB) yielded substantially worse results than a very simple link selection scheme based just on link loads (LLL+RL). Complex strategy (LDM+RM) based on both those data does not outperform LLL+RL substantially; moreover, their combination into LDM+RL hybrid gives best results.

Obviously, in the real network both metrics (traffic and number of links off) will be brought to the common denominator, i.e. the power consumption. Only then the single-criterion optimization problem emerges: the solution will be the network configuration guaranteeing most profitable balance of links savings and traffic transportation costs. As presented in the cited research, whatever the declared individual energy profiles of the equipment will be, there will be an ultimate stop criterion for the algorithm: stop if the physical power measurement system indicates that no further improvement can be done.

We believe that heuristics based on the traffic matrix still deserve attention and improvement. As regards RM, the most disappointing one, the choice of the link being activated must take more into account the effects it will make on links already present in Y_{BY}. Furthermore, all results reported here have been obtained on flow-level simulation, using Matlab computation environment with Biograph toolbox. Future works should perform verification by packet-level simulation or in a real testbed — also to check the impact of generated traffic on QoS parameters other than just bandwidth.

Although results the conclusions do not apply directly to the problem of energy effective routing in sensor networks, they can support the research in a number of ways. The algorithm can successively eliminate point-to-point node communication, which will lead to minimize the overall power consumption if energy model of a logical link is set appropriately. Moreover, the created simulation framework is a convenient basis for further research, more focused on the specifics of sensor communication.

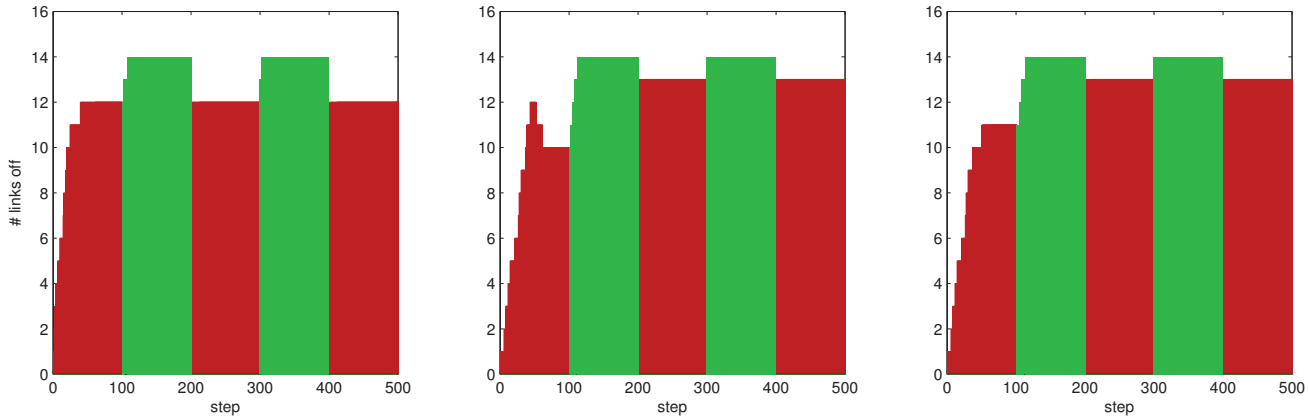
Acknowledgment

The work reported here was carried out with the support of Collaborative Project #258454 ECONET: Low Energy Consumption Networks, sponsored by the European Commission, and also with the support of Polish grant NN51 672940, sponsored by the National Science Centre.

References

[1] V. M. Akyildiz, I., *Wireless Sensor Networks.* John Wiley & Sons, Ltd., 2010.

[2] E. Niewiadomska-Szynkiewicz, "Energy aware communication protocols for wireless sensor networks," 2013, accepted by Transactions on Computational Collective Intelligence (Springer), issue 10 of TCCI, Vol. LNCS 7776.



(a) LLL+RL (b) LDM+RM (c) LDM+RL

Fig. 4. Number of links switched off in GÉANT network (graphs layout as in Fig. 3.)

[3] R. Bolla, R. Bruschi, A. Cianfrani, and M. Listanti, "Enabling backbone networks to sleep," *Network, IEEE*, vol. 25, no. 2, pp. 26–31, 2011.

[4] A. P. Bianzino, C. Chaudet, D. Rossi, and J.-L. Rougier, "A survey of green networking research," *Communications Surveys & Tutorials, IEEE*, vol. 14, no. 1, pp. 3–20, 2012.

[5] J. Wang, S. Ruepp, A. Manolova, L. Dittmann, S. Ricciardi, and D. Careglio, "Green-aware routing in gmpls networks," in *Computing, Networking and Communications (ICNC), 2012 International Conference on*, 2012, pp. 227–231.

[6] E. Niewiadomska-Szynkiewicz, A. Sikora, P. Arabas, and J. Kołodziej, "Control system for reducing energy consumption in backbone computer network," *Concurrency and Computation: Practice and Experience*, pp. n/a–n/a, 2012. [Online]. Available: http://dx.doi.org/10.1002/cpe.2964

[7] N. Vasić and D. Kostić, "Energy-aware traffic engineering," in *Proceedings of the 1st International Conference on Energy-Efficient Computing and Networking*. ACM, 2010, pp. 169–178.

[8] A. Cianfrani, V. Eramo, M. Listanti, and M. Polverini, "An OSPF enhancement for energy saving in ip networks," in *Computer Communications Workshops (INFOCOM WKSHPS), 2011 IEEE Conference on*. IEEE, 2011, pp. 325–330.

[9] A. P. Bianzino, L. Chiaraviglio, M. Mellia, and J.-L. Rougier, "GRiDA: Green distributed algorithm for energy-efficient IP backbone networks," *Computer Networks*, 2012.

[10] K.-H. Ho and C.-C. Cheung, "Green distributed routing protocol for sleep coordination in wired core networks," in *Networked Computing (INC), 2010 6th International Conference on*, 2010, pp. 1–6.

[11] M. Karpowicz, "Energy-effcient service rate control mechanisms for energy-aware processors: a design framework for the Linux kernel," Research and Acdemic Computer Network (NASK), Poland, Tech. Rep., 2013.

[12] V. Jarník, "O jistém problému minimálním," *Práca Moravské Prírodovedecké Spolecnosti*, vol. 6, pp. 57–63, 1930.

[13] J. B. Kruskal, "On the shortest spanning subtree of a graph and the traveling salesman problem," *Proceedings of the American Mathematical society*, vol. 7, no. 1, pp. 48–50, 1956.

[14] M. Kamola, "Traffic matrix estimation: Correlated flows case," submitted to IEEE Infocom 2014.

[15] A. Lakhina, K. Papagiannaki, M. Crovella, C. Diot, E. D. Kolaczyk, and N. Taft, "Structural analysis of network traffic flows," in *Proceedings of the joint international conference on Measurement and modeling of computer systems*, ser. SIGMETRICS '04/Performance '04. New York, NY, USA: ACM, 2004, pp. 61–72. [Online]. Available: http://doi.acm.org/10.1145/1005686.1005697

[16] I. Juva, "Sensitivity of traffic matrix estimation techniques to their underlying assumptions," in *Communications, 2007. ICC '07. IEEE International Conference on*, 2007, pp. 562–568.

[17] SndLib. Sndlib. [Online]. Available: http://sndlib.zib.de