Topology Control with Application Constraints

Michael Stein*, Géza Kulcsár†, Immanuel Schweizer*, Gergely Varró†, Andy Schürr† and Max Mühlhäuser*
Technische Universität Darmstadt, Germany
*{michael.stein, schweizer, max}@tk.informatik.tu-darmstadt.de
†{geza.kulcsar, gergely.varro, andy.schuerr}@es.tu-darmstadt.de

***Abstract*—Numerous topology control algorithms for wireless sensor networks exist. Typically, these algorithms optimize general network metrics like the transmission range of sensors. Being of general nature, they suit application-specific requirements insufficiently; e.g., application-specific communication patterns may favor links that the general algorithm will remove.**

We suggest a new research focus called application-specific topology control and propose a methodic approach for which we provide a first exploration. First, application-specific communication patterns are expressed as overlay graphs. Then, application-specific requirements are specified as constraints on this overlay. Next follows the core of the method, geared toward deriving application-specific topology control algorithms.

Applying the method to a data collection application with a many-to-one communication pattern, we propose two topology control algorithms. In a simulation study, we evaluate these algorithms against the existing algorithm kTC and a shortest path tree, showing that the method allows trading off general network optimization against application-specific optimization.

***Index Terms*—topology control, WSN, application-specific**

I. Introduction

Many topology control algorithms have been presented during the last decade [1]. Today, topology control algorithms are designed for all-to-all communication, thus, optimizing general network metrics like the transmission range of the nodes. However, wireless sensor networks (WSNs) are application oriented [2]. Most applications in WSNs follow specific communication patterns, e.g., the many-to-one pattern with all nodes sending data to a common base station. The corresponding application-specific communication overlays are not considered for topology control decisions in even the most practical algorithms. Hence, application-specific communication links might be removed, leading to severe implications for application performance.

Application-specific topology control has been identified as future work already in 2008 by Wang [1]. Xing et al. [3] propose the CTC algorithm, emphasizing that the required level of topology quality in a WSN depends on the application. CTC can be configured accordingly. However, CTC is unaware of the specific communication patterns introduced by different applications. The paper at hand is the first to explore application-specific topology control in depth. We model the application as an overlay graph to describe application-specific communication patterns. Application requirements can then be described as constraints on the overlay. Moreover, we propose a generic method to derive application-specific topology control algorithms from conventional topology control algorithms.

As a case study, we focus on the prevalent data collection application, which relies on the many-to-one communication pattern. Applying the overlay model and the generic method, we propose two application-specific topology control algorithms based on the existing `kTC` [4] algorithm.

We conduct a simulative evaluation of these algorithms against `kTC` and a shortest path tree as baseline. The evaluation illustrates that our application-specific algorithms preserve application-specific links and are still able to optimize the network for general communication. Demonstrating the trade-off between application-specific optimization and general optimization, we show that our new algorithms can be configured to control this trade-off as desired.

II. System Model

In wireless networks, all nodes typically send with maximum transmission power, reaching as many nodes as possible. Employing topology control, each node reduces the number of neighbors such that the resulting topology is still connected and has beneficial properties. This edge reduction allows nodes to reduce their transmission power, hence, saving energy.

Topology control is usually employed on the underlay only. The underlay is an undirected graph $G_U = (V, E_U)$ in which two nodes $u, v \in V$ are connected by an edge $(u, v) \in E_U$ if there is a physical link between them. An example of an underlay is depicted in Fig. 1 by black solid lines.

A WSN is usually deployed to serve one application [2]. An application induces an overlay $G_O = (V, E_O)$, which reflects the communication pattern of the application. An example of an overlay is depicted in Fig. 1 by dashed lines.

The contribution of this paper builds upon the observation that topology control increases distances on the overlay, thus, affecting application performance. This can be observed in Fig. 1, where four nodes (v_1, v_2, v_3, v_4) send data to a base station (b). After execution of the `kTC` algorithm, which removes two edges e_1 and e_2 from the graph, the hop count to b increases for all nodes but for v_3.

In order to provide good application performance, topology control must ensure that the distance increase in the overlay is sufficiently small. Based on the underlay G_U, each overlay edge $(u, v) \in E_O$ has a distance weight denoted by $\mathcal{D}_{\mathcal{M}}(G_U, u, v)$, where $\mathcal{M} : E_U \to \mathcal{R}$ is an overlay distance metric, which assigns an application-specific routing weight to underlay edges. $\mathcal{D}_{\mathcal{M}}(G_U, u, v)$ is the minimum distance over all paths between u and v in G_U. For a path $v_1 ... v_n$, we define the distance as the sum of the underlay edge weights.

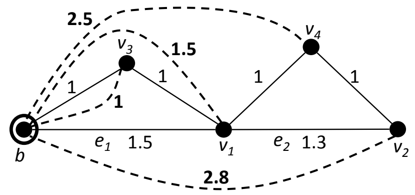


Fig. 1. Example topology with Euclidean edge weights.

Typically, the overlay distance metric $\mathcal{M}$ is defined based on the application requirements. For example, applications relying on strict delay constraints, like factory automation systems [5], mainly use the number of hops between nodes to specify the costs of routing paths. Other applications aim at high energy efficiency. These applications specify the cost of each link based on the associated energy consumption. For the sake of simplicity, we use the Euclidean distance between two nodes to specify the cost of the edge $e \in E_U$, i.e., $\mathcal{M}(e) = ||e||$.

III. Application-specific Topology Control

This section presents our approach to application-specific topology control. First, we provide a generic method. This method is then applied to the many-to-one communication pattern, which is found in most WSN applications.

We aim at achieving both sufficient general network performance and good application performance. Therefore, we suggest to execute a conventional topology control algorithm, but define application constraints to limit the edge removal to edges that are not specific to the application. The corresponding method consists of three sequentially conducted phases:

Phase I: Edge marking. A conventional topology control algorithm is executed. However, instead of removing edges immediately, the edges are *marked* for further processing.

Phase II: Removal candidate edge selection. Based on the set of marked edges, an edge selection strategy selects a subset of *removal candidate edges*. This strategy considers graph constraints, which reflect application-specific requirements.

Phase III: Edge removal. Only the removal candidate edges are then removed from E_U.

In the following, $E^m \subseteq E_U$ and $E^r \subseteq E^m$ denote the sets of marked edges and removal candidate edges, respectively.

This is a generic method, hence, there is some degree of freedom. First, a topology control algorithm has to be selected for the execution in Phase I. Second, an edge selection strategy needs to be specified. For the second aspect, we suggest to model both the network and the application as graphs as done in Section II. This allows for defining application-specific constraints in an efficient way. These constraints can then be used to develop the edge selection strategy.

For the rest of this paper, we will focus on the many-to-one communication pattern. This communication pattern is often used in WSN applications because sensor nodes typically report collected data to a central base station. We will first model the data collection application as an overlay and define a corresponding application constraint. Afterwards, we will select an algorithm for the execution in Phase I, and provide two edge selection strategies to ensure this constraint.

We model the data collection application as follows.

Definition 1. *For the data collection application, the overlay is an undirected graph $G_O = (V, E_O)$ in which all nodes $v_i \in V \setminus \{b\}$ are connected to the base station b by an overlay edge $e_i \in E_O$.*

An example of such an overlay is again depicted in Fig. 1.

As noted before, we preserve application-specific edges by introducing constraints on the overlay. For the many-to-one communication pattern, we specify the following constraint.

Constraint 1. *$G_U^{old} = (V, E_U)$ and $G_U^{new} = (V, E_U \setminus E^r)$ are the underlay graphs before and after the execution of topology control, respectively. For a base station $b \in V$, an overlay distance metric $\mathcal{M}$ and a stretch parameter $a \in [1, \infty[$,*

$$\forall v \in V : \frac{\mathcal{D}_{\mathcal{M}}(G_U^{new}, v, b)}{\mathcal{D}_{\mathcal{M}}(G_U^{old}, v, b)} \leq a.$$

A low overlay distance is assumed to indicate good application performance. Constraint 1 specifies that the overlay distance from each node to the base station must not increase by more than a given stretch parameter a.

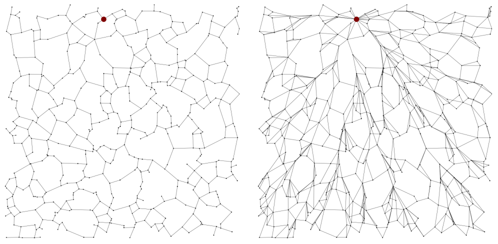
In order to mark edges in Phase I, we use `kTC` [4], a state of the art topology control algorithm. `kTC` removes the longest edge of each triangle in G_U if it is at least k times longer than the shortest edge in the triangle. We denote the resulting underlay by $G_{\texttt{kTC}}$. It is important to note that any other topology control algorithm can be used to mark edges.

The challenging part of our proposed method is to provide a suitable edge selection strategy. The main difficulty of edge selection is the huge solution space: As the edge selection strategy may output any subset $E^r \subseteq E^m$, there are in total $2^{|E^m|}$ possible selections. Moreover, the removal decisions for different edges are inter-dependent because a path on the overlay typically employs multiple edges on the underlay. In other words, the decision if a certain edge can be removed without violating the overlay constraints typically depends on whether other marked edges are removed as well.

There are two intuitive assumptions that can be made when making the removal decision for an edge $e \in E^m$. We can either assume that the other marked edges $E^m \setminus e$ are preserved, or we can assume that these edges are removed. These assumptions reflect the core ideas of our two edge selection algorithms, which are presented in the following.

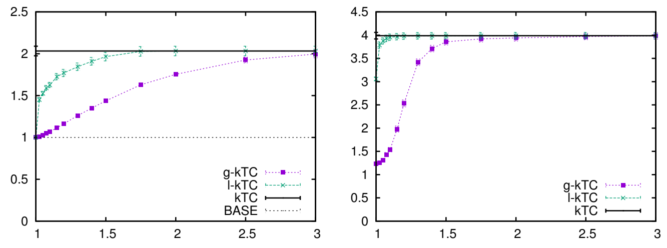
A. `l-kTC`

The first edge selection strategy, `local-kTC` (`l-kTC`), builds upon the assumption that none of the marked edges but the investigated local one will be removed in Phase III. First, the algorithm computes the overlay distance $\mathcal{D}_{\mathcal{M}}(G_U, v, b)$ from each node $v \in V$ to the base station b. For each marked edge, `l-kTC` temporarily removes the edge from the underlay. Then, it updates the distances of the incident nodes to b and re-adds the edge to the underlay. If the increase in distance for the incident nodes is less than or equal to the stretch parameter a,



(a) kTC (b) l-kTC ($a = 1.0$)

Fig. 2. kTC and l-kTC topologies. The base station b is marked red.



(a) Base stretch (b) General stretch

Fig. 3. Length stretch factors (x-axis: stretch parameter a)

the edge becomes a removal candidate. The resulting underlay after Phase III is denoted by $G_{\texttt{l-kTC}}$.

For example, executing l-kTC ($a = 1.4$) for the topology shown in Fig. 1 would confirm the removal of both e_1 and e_2.

B. g-kTC

The second edge selection strategy, global-kTC (g-kTC), differs from l-kTC regarding the assumption being made about the removal decisions of other marked edges. When g-kTC investigates the implications of the removal of an edge $e \in E^m$, it expects that globally all other marked edges will be removed as well. First, just like l-kTC, g-kTC computes the overlay distance from each node to the base station. Then, the algorithm temporarily removes all marked edges from the underlay. Next, g-kTC iterates over all marked edges, recomputes the overlay distances for the incident nodes and adds an edge to the set of removal candidates only if the increase in overlay distance is less than the stretch parameter a for both incident nodes. Finally, the algorithm re-adds all marked edges to the underlay. The resulting underlay after Phase III is denoted by $G_{\texttt{g-kTC}}$.

For example, executing g-kTC ($a = 1.4$) for the topology shown in Fig. 1 would remove e_2 and preserve e_1.

IV. Theoretical Analysis

This section provides a theoretical analysis of the algorithms introduced in Section III. A key result is that $G_{\texttt{kTC}}$ is a subgraph of both $G_{\texttt{l-kTC}}$ and $G_{\texttt{g-kTC}}$. This subgraph relation has two important implications. First, as $G_{\texttt{kTC}}$ is connected [4], $G_{\texttt{l-kTC}}$ and $G_{\texttt{g-kTC}}$ are connected. Second, each path that exists in $G_{\texttt{kTC}}$ does also exist in $G_{\texttt{l-kTC}}$ and $G_{\texttt{g-kTC}}$. Thus, the distances between arbitrary nodes in $G_{\texttt{l-kTC}}$ or $G_{\texttt{g-kTC}}$ are bounded by those in $G_{\texttt{kTC}}$. Additionally, we can prove for g-kTC by induction[1] that Constraint 1 is fulfilled.

While l-kTC and g-kTC retain connectivity and symmetry from kTC, it is also important to state that $G_{\texttt{l-kTC}}$ and $G_{\texttt{g-kTC}}$ are neither degree-bounded nor planar, while $G_{\texttt{kTC}}$ is [4]. Moreover, in contrast to kTC [4], $G_{\texttt{l-kTC}}$ and $G_{\texttt{g-kTC}}$ are not θ-separated, i.e., it cannot be shown that two logical neighbors are separated by at least an angle of θ. This means, l-kTC and g-kTC lose some of the theoretical properties provided by kTC. However, we argue that the application specifies the key requirements for topology control. If a theoretical property is important for the application, this should be encoded into the topology control constraints provided by the application.

V. Evaluation

This section evaluates our approach to application-specific topology control. We state three main hypotheses:

1) l-kTC and g-kTC promise better application performance than kTC in terms of overlay distance between the sensor nodes and the base station.
2) Building on the general algorithm kTC, l-kTC and g-kTC still achieve good general network performance.
3) There exists a trade-off between general performance and application-specific performance, which can be controlled with the stretch parameter a.

In order to evaluate these hypotheses, we conducted a simulation study, which is presented in the following.

A. Simulation Setup

We used an integrated simulation environment based on the network simulator PeerfactSim.KOM [6] to generate topologies and the graph transformation tool eMoflon [7] to execute topology control[2]. We simulated 500 nodes including one base station. The nodes are placed uniformly at random onto a squared plane with a side length of 10 units, and each node has a maximum transmission range of 1 unit. Each simulation setup was executed 200 times with different random seeds. All results are given with 95% confidence intervals.

We compare l-kTC and g-kTC to kTC and the UDG. kTC was always executed with $k = 1.0$. Additionally, we consider the most intuitive application-specific topology, called BASE, which is a shortest path tree rooted at the base station.

B. Hypothesis 1: Application Performance

In our case study, we focus on overlays with the many-to-one communication pattern (cf. Definition 1). Thus, the stretch factor for the overlay distance from the sensor nodes to the base station is the most important metric to measure the application performance. In the following, we denote the stretch factor that reflects the maximum increase in distance over all $|V| - 1$ sensor nodes to the base station by the term

[1]The proof has been omitted due to space restrictions.

[2]For more details about graph transformation and the corresponding model, please refer to [8].

base stretch factor. The base stretch factors for the different topologies are given in Fig. 3a.

BASE reflects the application-specific optimum. Creating a shortest path tree rooted at the base station, BASE has a base stretch factor of 1. On the other hand, kTC is application-agnostic, resulting in the removal of many edges (cf. Fig. 2a) and a base stretch factor of slightly above 2. It must be expected that this increase in path length degrades the application performance significantly compared to the UDG.

l-kTC and g-kTC build upon kTC but also consider Constraint 1. Consequently, both algorithms promise better application performance than kTC. For a small stretch parameter a, l-kTC and g-kTC achieve a stretch factor close to 1. Increasing a relaxes the overlay constraint, which leads to the removal of more application-specific edges and, thus, to an increase of the base stretch factor. Nevertheless, as $G_{\mathtt{kTC}}$ is a subgraph of $G_{\mathtt{l-kTC}}$ and $G_{\mathtt{g-kTC}}$, their stretch factors are bounded by the ones of kTC. Moreover, g-kTC even guarantees that the base stretch factor is bounded by a.

C. Hypothesis 2: General Performance

For general network performance, it is crucial to provide short paths for all-to-all communication. We measure the *general stretch factor*, which gives the maximum increase in path length between all $|V|(|V|-1)/2$ pairs of nodes. Fig. 3b gives the general stretch factors of kTC, l-kTC and g-kTC.

As kTC strives for general network optimization, it also provides a reasonable general stretch factor. BASE creates a tree topology that is solely optimized for the data collection application. Having a general stretch factor of 205.16, the resulting topology is inappropriate for all-to-all communication.

Additionally, we notice that the general stretch factors of l-kTC and g-kTC are slightly smaller than the ones of kTC. The reason for this is that the preserved application-specific edges may also be used for all-to-all communication.

A wireless node may shrink its transmission power such that the node still reaches its farthest neighbor. The *average maximum transmission range* gives the average Euclidean distance from nodes to their farthest neighbor. A small value is beneficial because the required transmission power increases with distance. Also, small transmission power reduces network contention. For these reasons, we consider transmission range reduction to be the main target of topology control.

All topology control algorithms achieve significant transmission range reductions (Fig. 4). For the initial UDG, the average maximum transmission range is slightly above 0.96. kTC reduces this value to about 0.46. This reflects a reduction of more than 50% compared to the UDG. BASE achieves a transmission range reduction of around 25%.

Recall that $G_{\mathtt{kTC}}$ is a subgraph of $G_{\mathtt{l-kTC}}$ and $G_{\mathtt{g-kTC}}$. Therefore, g-kTC and l-kTC cannot outperform kTC in terms of transmission range reduction. Fig. 2b demonstrates that l-kTC preserves only those edges that point toward the base station, creating an intuitive visual pattern. Consequently, even for a small stretch parameter ($a \leq 1.1$), the transmission range reduction is very close to the one of kTC. This is unfortunately not the case for g-kTC, which requires strong relaxation of Constraint 1 for high transmission range reduction.

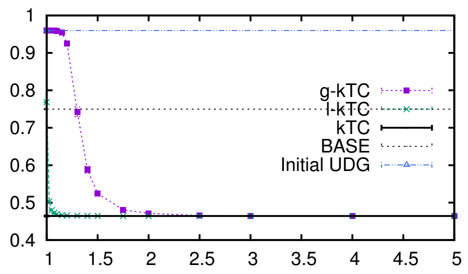


Fig. 4. Average maximum transmission range (x-axis: stretch parameter a)

D. Hypothesis 3: Trade-off

kTC and BASE are the extreme cases that focus solely on general network performance and application performance, respectively. Consequently, neither kTC nor BASE perform well for both. It becomes evident that application-specific performance and general performance are competing optimization goals: They constitute a trade-off. l-kTC and g-kTC aim at balancing this trade-off. As both algorithms build upon a conventional algorithm but consider application constraints, l-kTC and g-kTC are able to optimize the application without making all-to-all communication infeasible.

For l-kTC and g-kTC, the stretch parameter a adjusts the aggressiveness. Low aggressiveness leads to an application-specific topology, whereas high aggressiveness improves general network performance. Consequently, we are able to control the trade-off above as desired.

In summary, we explored application-specific topology control. As the evaluation showed, this is possible with a trade-off between application-specific and general performance.

Acknowledgment

This work has been funded by the German Research Foundation (DFG) as part of project A01 within the Collaborative Research Center (CRC) 1053 – MAKI.

References

[1] Y. Wang, "Topology Control for Wireless Sensor Networks," in *Wireless Sensor Networks and Applications*, 2008, pp. 113–147.
[2] J. Yick, B. Mukherjee, and D. Ghosal, "Wireless sensor network survey," *Computer Networks*, vol. 52, no. 12, pp. 2292–2330, 2008.
[3] G. Xing, C. Lu, X. Jia, and R. Pless, "Localized and configurable topology control in lossy wireless sensor networks," *Ad Hoc Networks*, vol. 11, no. 4, pp. 1345–1358, 2013.
[4] I. Schweizer, M. Wagner, D. Bradler, M. Mühlhäuser, and T. Strufe, "kTC - Robust and Adaptive Wireless Ad-hoc Topology Control," in *ICCCN*, 2012, pp. 1–9.
[5] A. Kumar Somappa, K. Øvsthus, and L. Kristensen, "An Industrial Perspective on Wireless Sensor Networks – A Survey of Requirements, Protocols, and Challenges," *Communications Surveys Tutorials*, vol. 16, no. 3, pp. 1391–1412, 2014.
[6] D. Stingl, C. Gross, J. Ruckert, L. Nobach, A. Kovacevic, and R. Steinmetz, "PeerfactSim.KOM: A Simulation Framework for Peer-to-Peer Systems," in *HPCS*, 2011, pp. 577–584.
[7] eMoflon webpage. [Online]. Available: http://www.emoflon.org/
[8] G. Kulcsár, M. Stein, I. Schweizer, G. Varró, M. Mühlhäuser, and A. Schürr, "Rapid Prototyping of Topology Control Algorithms by Graph Transformation," in *GraBaTs*, 2014, pp. 1–14.