Amotz Bar-Noy
Magnús M. Halldórsson (Eds.)

LNCS 7718

Algorithms for Sensor Systems

8th International Symposium on Algorithms for Sensor Systems, Wireless Ad Hoc Networks and Autonomous Mobile Entities, ALGOSENSORS 2012 Ljubljana, Slovenia, September 2012, Revised Selected Papers

Lecture Notes in Computer Science 7718

Commenced Publication in 1973
Founding and Former Series Editors:
Gerhard Goos, Juris Hartmanis, and Jan van Leeuwen

Amotz Bar-Noy Magnús M. Halldórsson (Eds.)

Algorithms for Sensor Systems

8th International Symposium on Algorithms
for Sensor Systems, Wireless Ad Hoc Networks
and Autonomous Mobile Entities, ALGOSENSORS 2012
Ljubljana, Slovenia, September 13-14, 2012
Revised Selected Papers

Springer

Volume Editors

Amotz Bar-Noy
City University of New York
Department of Computer Science
365 Fifth Avenue
New York, NY 10016, USA
E-mail: amotz@sci.brooklyn.cuny.edu

Magnús M. Halldórsson
Reykjavik University
School of Computer Science
Menntavegur 1
101 Reykjavik, Iceland
E-mail: mmh@ru.is

ISSN 0302-9743 e-ISSN 1611-3349
ISBN 978-3-642-36091-6 e-ISBN 978-3-642-36092-3
DOI 10.1007/978-3-642-36092-3
Springer Heidelberg Dordrecht London New York

Library of Congress Control Number: 2012955409

CR Subject Classification (1998): F.2.1-2, G.1.2, G.2.2, C.2.0-1, C.2.3-6, I.2.9

LNCS Sublibrary: SL 5 – Computer Communication Networks
and Telecommunications

Typesetting: Camera-ready by author, data conversion by Scientific Publishing Services, Chennai, India

Printed on acid-free paper

Springer is part of Springer Science+Business Media (www.springer.com)

Preface

Wireless ad hoc sensor networks have recently become a very active research subject because of their high potential of providing diverse services to numerous important applications, including remote monitoring and tracking in environmental applications and low-maintenance ambient intelligence in everyday life. The effective and efficient realization of such large-scale, complex ad-hoc networking environments requires intensive, coordinated technical research and development efforts, especially in power-aware, scalable, robust wireless distributed protocols, owing to the unusual application requirements and the severe resource constraints of the sensor devices. On the other hand, a solid foundational background seems necessary for sensor networks to achieve their full potential. It is a challenge for abstract modeling, algorithmic design and analysis to achieve provably efficient, scalable, and fault-tolerant realizations of such huge, highly dynamic, complex, nonconventional networks. Features including the extremely large number of sensor devices in the network, the severe power, computing, and memory limitations, their dense, random deployment and frequent failures, pose new interesting abstract modeling, algorithmic design, analysis and implementation challenges of great practical impact. ALGOSENSORS aims to bring together research contributions related to diverse algorithmic and complexity theoretic aspects of wireless sensor networks.

Starting in 2011, ALGOSENSORS broadened its thematic scope, keeping its focus on sensor networks, but also including other related types of ad hoc wireless networks, such as mobile networks, radio networks, and distributed systems of robots. Papers are solicited into two tracks, one on Sensor Networks (Track A) and one on Ad Hoc Wireless and Mobile Systems (Track B). Furthermore, the status of the event has been upgraded to a symposium and its length extended to two days. ALGOSENSORS 2012, the 8th International Symposium on Algorithms for Sensor Systems, Wireless Ad Hoc Networks and Autonomous Mobile Entities, was held in Ljubljana, Slovenia, during September 13–14, 2012.

In 2012, there were 24 submissions to ALGOSENSORS: 14 to track A and ten to track B. After a careful selection procedure by the (joint) Program Committee (involving at least four reviews for each paper and five reviews for the vast majority papers, and fruitful discussions), 11 papers were accepted as full papers: five of them from track A and six of them from track B. In addition, two papers from track B were accepted as brief announcements. This volume contains these papers as well as summaries of the two keynote talks.

The five papers in Track A (Sensor Networks) present original research on topics such as barrier resilience, localization, connectivity with directional antennas, broadcast scheduling, and data aggregation. The topics covered by the

six papers in Track B (Ad hoc Wireless and Mobile Systems) include the SINR model, geometric routing, cognitive radio networks, video delivery, and mapping polygons.

We would like to warmly thank the ALGO/ESA 2012 organizers for kindly accepting the proposal of the Steering Committee to co-locate ALGOSENSORS with some of the leading events on algorithms in Europe. Also, we thank the keynote speakers Subhash Suri and Thomas Kesselheim for accepting our invitation. Many thanks go to the Program Committee members for their dedicated contribution toward a strong program.

October 2012

Amotz Bar-Noy
Magnús M. Halldórsson

Organization

Steering Committee

Josep Diaz	Polytechnic University of Catalonia, Spain
Bhaskar Krishnamachari	University of Southern California, USA
P.R. Kumar	University of Illinois, Urbana-Champaign, USA
Jan van Leeuwen	University of Utrecht, The Netherlands
Sotiris Nikoletseas	University of Patras and CTI, Greece
Jose Rolim	University of Geneva, Switzerland
Paul Spirakis	University of Patras and CTI, Greece

Program Committee

Track A Chair:	Amotz Bar-Noy	City University of New York, USA
Track B Chair:	Magnús M. Halldórsson	Reykjavik University, Iceland

Nikhil Bansal	Eindhoven Institute of Technology, The Netherlands
Prithwish Basu	BBN Technologies, USA
Shlomi Dolev	Ben Gurion University, Israel
Leah Epstein	University of Haifa, Israel
Thomas Erlebach	University of Leicester, UK
Guy Even	Tel Aviv University, Israel
Sándor Fekete	University of Technology, Braunschweig, Germany
Pierre Fraigniaud	CNRS and University of Paris Diderot, France
Jie Gao	SUNY Stony Brook, USA
Ramesh Govindan	University of South California, USA
Samir Khuller	University of Maryland, USA
Danny Krizanc	Wesleyan University, USA
Fabian Kuhn	University of Lugano, Switzerland
Pekka Orponen	Aalto University, Finland
Marina Papatriantafilou	Chalmers University of Technology, Sweden
Sriram Pemmaraju	University of Iowa, USA
Yvonne Anne Pignolet	ABB Research Laboratory, Switzerland
Geppino Pucci	Universitá di Padova, Italy
Dror Rawitz	Tel Aviv University, Israel
Christian Scheideler	Universität Paderborn, Germany
Stefan Schmid	Technische Universität Berlin, Germany
Mani Srivastava	UCLA, USA
Jukka Suomela	University of Helsinki, Finland

Subhash Suri	UCSB, USA
Takeshi Tokuyama	Tohoku University, Japan
Amy Y. Wang	Tsinghua University, China

Additional Referees

Benny Applebaum	Thomas Kesselheim	Xi Ming Li
Sebastian Daum	Hamed Khanmirza	Adrian Ogierman
Michael Dinitz	Matias Korman	Alberto Pettarin
Stefan Dobrev	Andreas Koutsopoulos	Andrea Pietracaprina
Carlo Fantozzi	Alexander Kröller	Moni Shahar
Vincenzo Gulisano	Olaf Landsiedel	Francesco Silvestri
Henning Hasemann	Asaf Levin	Sriram Venkateswaran
Csanád Imreh	Ximing Li	Stephan Wenger

Table of Contents

Approximation Algorithms for Wireless Spectrum Allocation with Power Control 1
Thomas Kesselheim

Geometric Computing over Uncertain Data 4
Subhash Suri

Packing Resizable Items with Application to Video Delivery over Wireless Networks 6
Sivan Albagli-Kim, Leah Epstein, Hadas Shachnai, and Tami Tamir

Symmetric Connectivity with Directional Antennas 18
Rom Aschner, Matthew J. Katz, and Gila Morgenstern

Comparative Study of Approximation Algorithms and Heuristics for SINR Scheduling with Power Control 30
Lukas Belke, Thomas Kesselheim, Arie M.C.A. Koster, and Berthold Vöcking

Approximating Barrier Resilience for Arrangements of Non-identical Disk Sensors 42
David Yu Cheng Chan and David Kirkpatrick

A Unified View to Greedy Geometric Routing Algorithms in Ad Hoc Networks 54
Jinhee Chun, Akiyoshi Shioura, Truong Minh Tien, and Takeshi Tokuyama

Big Data Interpolation an Efficient Sampling Alternative for Sensor Data Aggregation (Extended Abstract) 66
Hadassa Daltrophe, Shlomi Dolev, and Zvi Lotker

Mapping a Polygon with Holes Using a Compass 78
Yann Disser, Subir Kumar Ghosh, Matúš Mihalák, and Peter Widmayer

Metric Dimension for Gabriel Unit Disk Graphs Is NP-Complete 90
Stefan Hoffmann and Egon Wanke

Pseudo-scheduling: A New Approach to the Broadcast Scheduling Problem 93
Shaun N. Joseph and Lisa C. DiPippo

Self-stabilizing TDMA Algorithms for Dynamic Wireless Ad-Hoc Networks 105
Pierre Leone and Elad Michael Schiller

Complexity of Connectivity in Cognitive Radio Networks through Spectrum Assignment 108
Hongyu Liang, Tiancheng Lou, Haisheng Tan, Amy Yuexuan Wang, and Dongxiao Yu

On Some Bounds on the Optimum Schedule Length in the SINR Model 120
Tigran Tonoyan

Polynomial Time Approximation Algorithms for Localization Based on Unknown Signals 132
Johannes Wendeberg and Christian Schindelhauer

Author Index 145

Approximation Algorithms for Wireless Spectrum Allocation with Power Control

Thomas Kesselheim*

Department of Computer Science, RWTH Aachen University

Radio spectrum is one of the most valuable resources necessary to run wireless networks. Unfortunately, somewhat wasteful allocation has made it very scarce nowadays. Therefore, for future wireless evolution, we need smart algorithms that manage wireless spectrum access inside and between networks.

Often algorithmic studies dealing with optimization in wireless networks have modeled interference essentially by a graph. This simplified view of interference neglects a number of important aspects. Therefore, attention has recently shifted to more realistic models, in which interference constraints are stated based on the signal-to-interference-plus-noise ratio (SINR). These models allow to take power control into account: By adjusting transmit powers individually, one can significantly reduce the effects of interference.

1 Capacity-Maximization Problem

The most fundamental combinatorial optimization problem with power control is the following *capacity-maximization problem*: Given n pairs of senders and receivers (links), the task is to select a subset of these links and for each selected link a transmit power such that the number of successful transmissions is maximized.

One of the easiest ways to tackle this problem is to ignore the possibility of power control by simply set all transmit powers to a common, uniform value, and apply a suitable selection algorithm. Maybe surprisingly, this already yields approximation factors as good as $O(\log \Delta)$ [1], where Δ is the ratio between the maximum and the minimum distance between a sender and its corresponding receiver. Improved performances of $O(\log \log \Delta)$ can be reached by setting transmit powers proportional to the square-root of the distance between the senders and its receiver and applying a suitable section algorithm [3,4,2].

However, both mentioned bounds are tight: For both uniform and square-root power assignments, there are instances with ($\Delta = 2^n$ resp. $\Delta = 2^{2^n}$) in which one cannot get better approximation factors than $\Omega(n)$. Asymptotically, this is not even better than the trivial algorithm selecting only a single link, which has approximation factor n.

In [7], we show that using a different approach, a constant-factor approximation can be reached. In contrast to the algorithms mentioned above, it first selects the subset of links and assigns the transmit powers afterwards. Extensions of this

* This work has been supported by the UMIC Research Centre, RWTH Aachen University.

A. Bar-Noy and M.M. Halldórsson (Eds.): ALGOSENSORS 2012, LNCS 7718, pp. 1–3, 2013.

algorithm deal with limited transmit powers [9] rather than unlimited ones and with flexible data rates [8], in which the interference model

2 Secondary Spectrum Auctions

The insights from the considerations of the capacity-maximization problem can be reused in a more general context. In [6,5], we study a setting motivated by auctions for secondary spectrum markets. In these markets licenses allowing secondary-usage of currently unused parts of the spectrum are being sold. Licenses are valid for short periods of time and in local areas. Thus, they have to take interference into account, which we model by an edge-weighted graph. We extend the notion of independent sets to edge-weighted conflict graphs by requiring that the sum of incoming weights from all neighbors have to be less than 1 for all vertices in the set. By suitable choices of edge weights, the independent sets correspond the link sets for which there is a power assignment making all SINR constraints fulfilled.

Interestingly, the conflict graphs derived from the SINR model but also from a number of simpler interference models share a very important property. The *inductive independence number* ρ [10] is bounded by a constant or by a slowly growing function.

This property enables us to bypass the $\Omega(n^{1-\varepsilon})$ lower bound on the approximability of *maximum independent set* in general conflict graphs and get $O(\rho \cdot \log n)$-approximations for weighted maximum independent set edge-weighted conflict graphs. Also for the case of multiple channels, we devise approximation algorithms whose guarantees are almost optimal under standard complexity-theory assumptions. Furthermore, all approximation algorithms can be turned into truthful-in-expectation mechanisms ensuring that no bidder can benefit from lying about his true valuation.

References

1. Andrews, M., Dinitz, M.: Maximizing capacity in arbitrary wireless networks in the SINR model: Complexity and game theory. In: Proceedings of the 28th Conference of the IEEE Communications Society (INFOCOM), pp. 1332–1340 (2009)
2. Halldórsson, M.M.: Wireless Scheduling with Power Control. In: Fiat, A., Sanders, P. (eds.) ESA 2009. LNCS, vol. 5757, pp. 361–372. Springer, Heidelberg (2009)
3. Halldórsson, M.M., Holzer, S., Mitra, P., Wattenhofer, R.: The power of non-uniform wireless power. In: Proceedings of the 24th ACM-SIAM Symposium on Discrete Algorithms, SODA (to appear, 2013)
4. Halldórsson, M.M., Mitra, P.: Wireless capacity with oblivious power in general metrics. In: Proceedings of the 22nd ACM-SIAM Symposium on Discrete Algorithms (SODA), pp. 1538–1548 (2011)
5. Hoefer, M., Kesselheim, T.: Secondary spectrum auctions for symmetric and submodular bidders. In: Proceedings of the 13th ACM Conference on Electronic Commerce (EC), pp. 657–671 (2012)

6. Hoefer, M., Kesselheim, T., Vöcking, B.: Approximation algorithms for secondary spectrum auctions. In: Proceedings of the 23rd ACM Symposium on Parallelism in Algorithms and Architectures (SPAA), pp. 177–186 (2011)
7. Kesselheim, T.: A constant-factor approximation for wireless capacity maximization with power control in the SINR model. In: Proceedings of the 22nd ACM-SIAM Symposium on Discrete Algorithms (SODA), pp. 1549–1559 (2011)
8. Kesselheim, T.: Approximation Algorithms for Wireless Link Scheduling with Flexible Data Rates. In: Epstein, L., Ferragina, P. (eds.) ESA 2012. LNCS, vol. 7501, pp. 659–670. Springer, Heidelberg (2012)
9. Wan, P.-J., Ma, C., Tang, S., Xu, B.: Maximizing Capacity with Power Control under Physical Interference Model in Simplex Mode. In: Cheng, Y., Do Eun, Y., Qin, Z., Song, M., Xing, K. (eds.) WASA 2011. LNCS, vol. 6843, pp. 84–95. Springer, Heidelberg (2011)
10. Ye, Y., Borodin, A.: Elimination Graphs. In: Albers, S., Marchetti-Spaccamela, A., Matias, Y., Nikoletseas, S., Thomas, W. (eds.) ICALP 2009, Part I. LNCS, vol. 5555, pp. 774–785. Springer, Heidelberg (2009)

Geometric Computing over Uncertain Data

Subhash Suri

Department of Computer Science
University of California, Santa Barbara, CA 93106, USA

Geometric structures such as the convex hull, Delaunay triangulation, or minimum spanning tree (MST) are fundamental tools for reasoning about multi-dimensional data. What happens to these structures when the underlying data points are known with only partial certainty? For instance, what is the expected cost of the MST of a set of points, each known to be alive with some probability? Or, in a set of uncertain points, how likely is it that the closest pair is within distance L? This talk explores the effects of data uncertainty on the complexity of basic geometric problems.

Specifically, let us consider a set of points $M = \{s_1, s_2, \ldots, s_n\}$, called the *master* set, in a d-dimensional Euclidean space. Each point s_i is *active*, or present, with some independent and arbitrary but known (rational-valued) probability p_i. The independent point probabilities induce a sample space with 2^n outcomes, where an outcome $A \subseteq M$ occurs with probability $\Pr[S = A] = \prod_{s_i \in A} p_i \prod_{s_i \notin A} (1 - p_i)$.

We let $MST(A)$ denote the length of A's minimum spanning tree under the Euclidean norm. Let $S \subset M$ denote the set of active points in a trial, and let $MST(S)$ denote the *random variable* that assumes values $MST(A)$ over the 2^n subsets $A \subset M$. The expected length of the MST of S is the expectation of this random variable:

$$\mathbb{E}[MST(S)] \quad = \quad \sum_{A \subseteq M} \Pr[S = A] \cdot MST(A).$$

Despite the implicit summation over an exponential number of subsets, we observe that the expected value of many basic geometric structures can be computed easily. In particular, the expected perimeter or area of the convex hull of S, or the expected lengths of various proximity graphs including the Delaunay triangulation, Gabriel graph and the Relative Neighborhood graph are easily computed in polynomial time. In this context, it is a bit surprising that computing the expected length of the MST proves to be intractable. Our results on the stochastic MST include the following:

- computing $\mathbb{E}[MST(S)]$ is $\#P$-hard for any dimension $d \geq 2$; the problem is trivial for $d = 1$.
- a simple FPRAS (fully polynomial randomized approximation scheme) for approximating $\mathbb{E}[MST(S)]$ in a metric space, and thus also in a Euclidean space.

A. Bar-Noy and M.M. Halldórsson (Eds.): ALGOSENSORS 2012, LNCS 7718, pp. 4–5, 2013.

- an $O(n^4)$-time deterministic algorithm for approximating $\mathbb{E}[MST(S)]$ within a constant factor in two dimensions.
- a proof of hardness for estimating the tail bounds of the distribution of $MST(S)$. We show that this cannot be approximated to any *multiplicative factor* in a general metric space, assuming $P \neq NP$.

We also study the complexity of closest pair and nearest neighbor searching for these stochastic points, and show the following results:

- It is $\#P$-hard to compute the probability that the closest pair of points have distance at most a value ℓ, even for dimension 2 under the L_∞ norm.
- In the *linearly-separable* and *bichromatic* planar case, the closest pair probability can be computed in polynomial time under the L_∞ norm.
- Without the linear separability, even the bichromatic version of the stochastic closest pair problem is $\#P$-hard under the L_2 or L_∞ norm.
- Even with linear separability and L_∞ norm, the bichromatic case becomes $\#P$-hard in dimension $d \geq 3$.
- We give a linear-space data structure with $O(\log n)$ query time to compute the expected distance of a given query point to its $(1+\varepsilon)$-approximate nearest neighbor when the dimension d is a constant.

This talk is based on the following two papers:

- Stochastic Minimum Spanning Trees in Euclidean Spaces.
  Pegah Kamousi, Timothy Chan, and Subhash Suri.
  Proc. 27th Annual Symposium on Computational Geometry (SoCG) '11, Paris, France, June 13-15, 2011.
- Closest Pair and the Post Office Problem for Stochastic Points.
  Pegah Kamousi, Timothy Chan, and Subhash Suri.
  Proc. 12th International Symposium on Algorithms and Data Structures (WADS) '11, Brooklyn, NY, Aug 15-17, 2011.

Packing Resizable Items with Application to Video Delivery over Wireless Networks*

Sivan Albagli-Kim[1], Leah Epstein[2], Hadas Shachnai[1], and Tami Tamir[3]

[1] Computer Science Department, Technion, Haifa 32000, Israel
{sivanal,hadas}@cs.technion.ac.il
[2] Department of Mathematics, University of Haifa, Haifa, Israel
lea@math.haifa.ac.il
[3] School of Computer Science, The Interdisciplinary Center, Herzliya, Israel
tami@idc.ac.il

Abstract. Motivated by fundamental optimization problems in video delivery over wireless networks, we consider the following problem of *packing resizable items (PRI)*. Given is a bin of capacity $B > 0$, and a set I of items. Each item $j \in I$ is of size $s_j > 0$. A packed item must stay in the bin for a fixed time interval. To accommodate more items in the bin, each item j can be *compressed* to a size $p_j \in [0, s_j)$ for at most a fraction $q_j \in [0, 1)$ of the packing interval. The goal is to pack in the bin, for the given time interval, a subset of items of maximum cardinality. PRI is strongly NP-hard already for highly restricted instances.

Our main result is an approximation algorithm that packs, for any instance I of PRI, at least $\frac{2}{3}OPT(I)-3$ items, where $OPT(I)$ is the number of items packed in an optimal solution. Our algorithm yields better ratio for instances in which the maximum compression time of an item is $q_{max} \in (0, \frac{1}{2})$. For subclasses of instances arising in realistic scenarios, we give an algorithm that packs at least $OPT(I) - 2$ items. Finally, we show that a non-trivial subclass of instances admits an *asymptotic fully polynomial time approximation scheme (AFPTAS)*.

1 Introduction

Video content delivery over wireless networks is expected to grow exponentially in the coming years. It is driven by applications including streaming TV content to mobile devices, internet video, video on demand, personal video streaming, video sharing applications (from mobile to mobile), video conferencing, and live video broadcasting (cloud to mobile as well as mobile to cloud). In fact, a recent study (Cisco Visual Networking Index [3]) predicts that the mobile video traffic will be approximately two-thirds of the global mobile data traffic by 2015. Improvements in video compression and wireless spectral efficiency will not be

* Work partially supported by the Technion V.P.R. Fund, by Smoler Research Fund, and by the Ministry of Trade and Industry MAGNET program through the NEGEV Consortium (www.negev-initiative.org).

A. Bar-Noy and M.M. Halldórsson (Eds.): ALGOSENSORS 2012, LNCS 7718, pp. 6–17, 2013.

sufficient to accommodate this potential demand. This establishes the need for solutions on the intersection of theory and practice.

A common approach taken by companies, to better utilize the available bandwidth, is to deliver video content to the clients using different encodings. This enables the system to support mobile users, who tend to change location (or viewing devices) throughout the show. It also allows the system to degrade quality-of-service for bounded time intervals, while increasing the number of serviced clients. In wireless services, the available network bandwidth, shared by all users that are covered by an access point, is typically no more than 54Mbps. Therefore, no more than 36 MPEG-1 video streams can be delivered simultaneously to a local area [18]. This places strict limitation on the available bandwidth for simultaneous delivery of video content.

Consider a set of clients requesting to view video content over a wireless network. Suppose that each client is willing to tolerate a lower QoS level for some continuous time interval throughout the delivery.[1] The goal is to select a subset of the clients to be serviced and the QoS level for each client throughout the service, such that the total bandwidth allocated at any time does not exceed the available bandwidth, and the number of satisfied requests is maximized.

More specifically, given is a large database of video files, and a set of n clients. Suppose that, for some $q_j \in [0,1]$, client j is willing to view a fraction, q_j, of her requested video content in low QoS.[2] Each file is stored in the system in several encodings – corresponding to several levels of QoS. Assume that high QoS requires s_j bandwidth units, while a lower QoS level requires p_j bandwidth units, for some $0 < p_j < s_j$. Let B denote the total bandwidth available for file transmissions to the clients. The goal is to service the maximal number of clients, such that each client j receives high-QoS transmission, except maybe for the pre-agreed fraction q_j of the video show, in which the client may receive a lower QoS. The degradation in QoS transmission may occur at most once throughout the transmission of the video content to certain client, (i.e., along a *contiguous* segment of the transmitted content).

We model this optimization problem as the following problem of *packing resizable items (PRI)*. Given is a set I of n items and a bin of capacity $B > 0$. Each packed item must stay in the bin for a given time interval. Each item j is associated with a size $0 < s_j \leq B$ (also called expanded or non-compressed size), a compressed size $0 \leq p_j < s_j$, and a compression time, $0 \leq q_j < 1$, specifying the maximal fraction of the packing interval the item can be stored in its compressed size. The time interval in which item j is compressed must be contiguous. The goal is to pack in the bin a feasible subset of the items of maximum cardinality for the given time interval. W.l.o.g., we assume that the packing interval is $(0,1]$. Thus, a solution for *PRI* specifies the subset of packed items $I' \subseteq I$ and, for any $j \in I'$, the interval $(c_j, e_j]$ in which j is compressed,

[1] The continuity requirement comes from the fact that repeated changes in encoding of the transmitted content may cause the client unpleasant interruptions.

[2] Allowing such degradation in QoS reduces the rates for the clients.

such that the total size of expanded and compressed items at any time $t \in (0, 1]$ is at most B.

The above application of video delivery over wireless network yields a general instance of PRI. When all clients share the same low-QoS encoding, we get the special case of *PRI* with uniform compressed size (see Section 4). When all clients share the same pre-agreed low-QoS fraction, we get an instance of *PRI* with uniform compression time (see Section 3.1).

By our definition of PRI, in any feasible packing, item j is expanded during time interval $(0, c_j]$, compressed during the interval $(c_j, e_j]$, and expanded again during the interval $(e_j, 1]$ for some $0 \leq c_j \leq e_j \leq 1$, where $0 \leq e_j - c_j \leq q_j$. When expanded, the item consumes capacity s_j in the bin, while in its compressed form it consumes capacity p_j. W.l.o.g., we assume that $B < \sum_j s_j$, since otherwise, all items can be packed with no compression for the whole time interval.

In [1] we show that *PRI* is strongly NP-hard already for highly restricted instances, where all items have the same expanded and compressed sizes, or the same compression times. Indeed, solving PRI involves the selection of a subset of items to be packed, as well as finding a feasible placement for these items in the bin. This makes PRI harder than other single bin packing problems.

Figure 1 presents some examples of PRI instances, and their packings. Figure 1(a) presents an optimal packing of 6 items with uniform expanded size, where $s_j = 1$ for all j, and uniform compression time, i.e., $q_j = 1/3$ for all j. The compressed sizes are $0.2, 0.4, 0.5, 0.5, 0.6$, and 0.8, and the bin capacity is $B = 5$. Figure 1(b_1) presents an optimal packing of 6 items with uniform expanded size, i.e. $s_j = 1$ for j, and uniform compressed size $p_j = 1/3$ for all j. The compression times are $0.2, 0.4, 0.5, 0.5, 0.6$, and 0.8. As shown in the figures, parts of the same item can be stored at different heights in the bin, as long as the total capacity allocated to item j along its expanded and compressed intervals are s_j and p_j, respectively. These characteristics of the packings follow from the nature of our applications, in which the allocation of resource capacity (such as bandwidth, or servers on a cloud) to an element is not required to be contiguous.

Figure 1(b_2) demonstrates the challenge of finding a feasible placement for the selected items. The instance is the same as the one packed in Figure 1(b_1). A natural greedy approach is to pack the items one after the other, while balancing the load along the packing interval. Thus, each item is packed as compressed in a 'more loaded' sub-interval, and as expanded in other parts of the packing interval. As shown in Figure 1(b_2), packing the items using this approach requires a bin of capacity $B = 4\frac{2}{3}$. While the greedy approach yields efficient approximation in some cases, it is not well defined for arbitrary instances, since the more loaded sub-interval may not be contiguous.

Another difficulty in solving *PRI* comes from the fact that there is no natural *ordering* for the packed items. For example, in the Knapsack problem, it is well known that an optimal fractional solution consists of items with the highest profit/size ratios. Moreover, with unit profits, Knapsack can be trivially solved by packing the smallest items. In *PRI*, items have unit profits and can be ordered by their total demand for capacity (given by $s_j(1-q_j)+p_jq_j$), however, an optimal

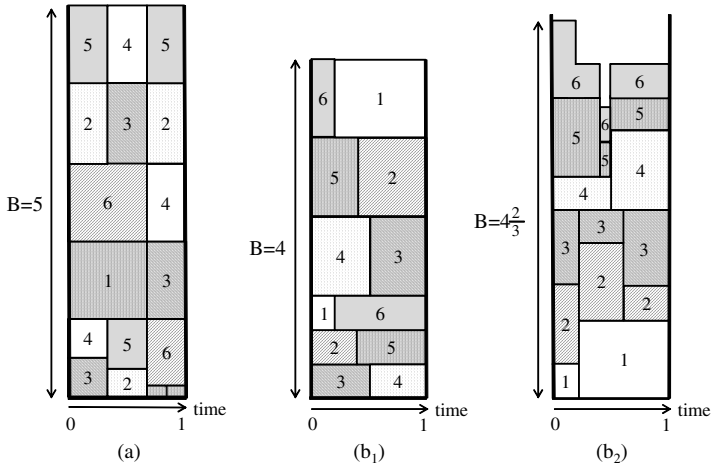


Fig. 1. (a) An optimal packing of an instance with uniform q. An optimal (b_1) and greedy (b_2) packing of an instance with uniform p.

solution does not necessarily pack the items having the smallest demands. This is valid even for a fractional solution, in which items may be partially packed (in a fractional solution the expanded and compressed sizes are divided).

1.1 Related Work

Packing items in a single bin, or in multiple bins, have been widely studied (see [12] for a survey). Most of these problems are NP-hard. The 0/1-Knapsack problem admits a *fully polynomial time approximation scheme (FPTAS)*, based on a pseudo-polynomial time solution. That is, for any $\varepsilon > 0$, a $(1-\varepsilon)$-approximation to the optimal can be found in $O(n/\varepsilon^2)$, where n is the number of items [16]. In contrast, the *multiple knapsack problem (MKP)* is NP-hard in the strong sense, therefore it is unlikely to have an FPTAS [15]. A PTAS for MKP was presented by Chekuri and Khanna [5].

Our problem of packing resizable items is closely related to the *bin covering (BC)* problem, that is well studied (see, e.g., [2,6,7,11]). The input for bin covering is a set of items $\{a_1, \ldots, a_n\}$; each item a_j has a size $s(a_j) \in (0, 1)$. The goal is to pack the items into unit sized bins so as to maximize the number of bins that receive items of total size at least 1. It is easy to see that no polynomial time algorithm can have approximation ratio better than $\frac{1}{2}$ (indeed, applying such approximation algorithm to instances where the total size of items is 2 would solve the Partition problem). The first *asymptotic approximation scheme (APTAS)* for bin covering was introduced by Csirik et al. [7]. Subsequently, Jansen and Solis-Oba [11] presented an AFPTAS for the problem. We discuss the relation between the two problems in Section 4.

In computational geometry, the problems of *covering a region by rectangles* [4,10,17] and *tiling by rectangles* [13] are well studied. Interestingly, these

problems also relate to *PRI* (see Section 4.1). However, in both problems, the rectangles can translate, but cannot move along the x- or y-axis. Therefore, the techniques used to solve these problems cannot be applied when solving PRI.

Other related work deal with efficient broadcasting over wireless networks. This combines technical as well as theoretical aspects, in particular, algorithms for efficient bandwidth allocation, streaming, and routing [9,8].

1.2 Our Contribution

We give a comprehensive study of PRI. For some subclasses of instances, our results are almost the best possible. Let $OPT(I)$ denote an optimal solution for an instance I. When clear from the context, we omit I. We use OPT also to denote the cardinality of $OPT(I)$. Our main result (in Section 2) is an approximation algorithm for general instances of PRI. The performance of the algorithm depends on q_{max}, the maximum compression time of any item. Specifically, let $\gamma = \left\lceil \frac{1}{q_{max}} \right\rceil - 1$. If the compression times may take any value in $[0, 1)$, in particular, if $q_{max} \geq \frac{1}{2}$, then $\gamma = 1$. In this case, the algorithm packs at least $\frac{2}{3}OPT - 3$ items. If for all j, $0 < q_j < \frac{1}{2}$, then $\gamma \geq 2$, and the algorithm packs at least $\frac{2\gamma+2}{2\gamma+3}OPT - 3$ items. For the case of uniform expanded size inputs, where $s_j = 1$ for all j, the algorithm packs at least $\frac{3}{4}OPT - 3$ items. We note that in the application motivating our work, all items tend to have small compression times, resulting in good approximation ratio. In particular, for $\gamma = 6$, a typical value in our application of video services, we obtain an asymptotic 14/15-approximation algorithm.

For other subclasses of instances arising in realistic scenarios, where compression times are drawn from a divisible sequence, we give (in Section 3) an algorithm that packs $OPT(I) - 2$ items. Furthermore, we show (in Section 4) that a non-trivial subclass of instances, of items with uniform compressed and expanded size, admits an *asymptotic fully polynomial time approximation scheme (AFPTAS)*.

Due to space constraints, some of the proofs are omitted. The detailed results appear in [1].

Techniques. In deriving our results for general PRI instances (in Section 2), we make non-standard use of a rounding technique applied in the Harmonic algorithm for Bin packing [14]. Our algorithm initially selects the subset of packed items; then, the compression times of the items are rounded down to one of three values, where each value is a unit fraction that depends on γ. We show how each of these subsets, which has uniform compression time, can be packed.

In developing (in Section 4) an AFPTAS for instances with uniform size, we use a transformation of PRI to the problem of covering a region by sliceable rectangles, which finds applications also in computational geometry. Our *covering with holes* technique (Section 4.1) enables to draw a non-trivial connection between PRI and the bin covering problem.

2 Approximation Algorithm for General Instances

In this section we present an approximation algorithm, denoted Alg_{arb}, for instances of PRI with arbitrary expanded sizes, s_j, and arbitrary compressed sizes $0 < p_j < s_j$. For an item j, let $r_j = s_j - p_j$, and let the *weight* of an item be $w_j = s_j(1-q_j) + p_j q_j = s_j - r_j q_j$, that is, w_j is the total capacity required for item j along the packing interval. For a set of items S, let $w(S) = \sum_{j \in S} w_j$, $s(S) = \sum_{j \in S} s_j$, $p(S) = \sum_{j \in S} p_j$, and $r(S) = \sum_{j \in S} r_j$.

The idea of Alg_{arb} is to select first the subset of packed items, then round down the compression time of each item into one of three values, and then pack each of the resulting sets (having uniform compression times) separately. Let I_{uni-q} be an instance with uniform compression time, such that for all $j \in I_{uni-q}$, $q_j = \frac{1}{\alpha}$. In Section 3.1, we present an almost optimal algorithm for such instances and show that it is possible to pack $|I_{uni-q}| - 1$ items in a bin of capacity $w(I_{uni-q})$. This algorithm is used as a subroutine by Alg_{arb}.

The next simple observation refers to instances in which items cannot be compressed at all.

Observation 1. *If $q_j = 0$ for all j, I can be packed in a bin of capacity $w(I)$.*

Algorithm Alg_{arb} partitions the items in I into three sets: $X = \{j \in I | \frac{1}{\gamma+1} \leq q_j < \frac{1}{\gamma}\}$, $Y = \{j \in I | \frac{1}{2(\gamma+1)} \leq q_j < \frac{1}{\gamma+1}\}$, and $Z = \{j \in I | 0 \leq q_j < \frac{1}{2(\gamma+1)}\}$. Let OPT be the set of items packed in an optimal solution, and let X_{opt}, Y_{opt}, Z_{opt} be the subsets of X, Y, Z respectively, in OPT.

Lemma 1. *If $\gamma \geq 2$ then $[(2\gamma)s(X_{opt}) + 3p(X_{opt})] + [(2\gamma+1)s(Y_{opt}) + 2p(Y_{opt})] + [(2\gamma+2)s(Z_{opt}) + p(Z_{opt})] \leq (2\gamma+3)B$, and if $\gamma = 1$ then $[s(X_{opt}) + 2p(X_{opt})] + [2s(Y_{opt} \cup Z_{opt}) + p(Y_{opt} \cup Z_{opt})] \leq 3B$.*

Lemma 2. *The total rounded weight of J^* is at most B.*

Algorithm Alg_{arb} receives an arbitrary PRI instance and proceeds as follows. In Steps 4 and 5, we use an almost optimal algorithm for instances with uniform compression times, given in Section 3.1.

$Alg_{arb}(I, B)$:

1. For each $j \in I$, round down q_j:
   a. If $j \in X$, let $q_j = \frac{1}{\gamma+1}$.
   b. If $j \in Y$, let $q_j = \frac{1}{2(\gamma+1)}$.
   c. If $j \in Z$, let $q_j = 0$.
2. Sort the items in I in a non-decreasing order according to their rounded weights w^r_j.
3. Let I' be the longest prefix in the sorted list having total rounded weight at most B. Let $X' = I' \cap X$, $Y' = I' \cap Y$, and $Z' = I' \cap Z$.
4. Pack $|X'| - 1$ items of X' in the bin using capacity $w^r(X')$.
5. Pack $|Y'| - 1$ items of Y' in the bin using capacity $w^r(Y')$.
6. Pack Z' in the bin using capacity $w^r(Z')$.

Lemma 3. *The algorithm outputs a feasible packing.*

Theorem 1. *If $\gamma \geq 2$, then Alg_{arb} returns a packing of at least $\frac{2\gamma+2}{2\gamma+3}OPT - 3$ items. If $\gamma = 1$, then Alg_{arb} returns a packing of at least $\frac{2}{3}OPT - 3$ items.*

Proof. We present the proof for $\gamma \geq 2$. The proof for $\gamma = 1$ is similar. In steps 4-6, the algorithm packs $|X'| - 1 + |Y'| - 1 + |Z'| = |I'| - 2$ items. We show that $|I'| \geq \frac{2\gamma+2}{2\gamma+3}OPT - 1$. Recall that I' form a prefix of the sorted list. Thus, the total rounded weight of any subset of at least $|I'| + 1$ items is larger than B. This implies that any set of items having total rounded weight at most B includes less than $|I'|+1$ items. In particular, by Lemma 2, as the total rounded weight of J^* is at most B, we have that $|I'| + 1 \geq |J^*|$ (note that J^* might include fractions and we consider here their fractional size). By definition of J^*, it includes $\frac{2\gamma+2}{2\gamma+3}OPT$ items. We conclude that $|I'| \geq \frac{2\gamma+2}{2\gamma+3}OPT - 1$. ∎

Uniform Expanded Sizes: In the case $\gamma = 1$, if for all items $s_j = 1$, Alg_{arb} returns a packing of at least $\frac{3}{4}OPT - 3$ items. To prove this, we can show that it is either the case that the set of $\frac{3}{4}|X|, \frac{3}{4}|Y|$ and $\frac{3}{4}|Z|$ items of minimum weight of the sets X, Y, Z, respectively, have total weight at most 1, or that the set of all items of X together with $\lfloor B - w(X) \rfloor$ additional items (that can always be packed), has a sufficiently large number of items. The bound $\frac{3}{4}$ is tight for this case, while the bound $\frac{2}{3}$ is tight for the general case (though the additive constant can be reduced to 2 by uniting Y and Z).

3 Almost Optimal Algorithm for Divisible Compression Times

In this section we present an almost optimal algorithm for instances in which the compression times form a divisible sequence.

Definition 1. *A sequence $\frac{1}{d_1} > \frac{1}{d_2} > \cdots > \frac{1}{d_z}$ is* divisible *if for all $1 \leq i \leq z$, d_i is an integer, and for all $1 \leq i \leq z-1$, d_{i+1} divides d_i.*

For example, $\frac{1}{2}, \frac{1}{4}, \frac{1}{8}, \frac{1}{16}, \frac{1}{32}$ and $\frac{1}{3}, \frac{1}{9}, \frac{1}{63}, \frac{1}{126}, \frac{1}{504}$ are divisible sequences. Let I be a PRI instance, in which item j has an arbitrary expanded size, s_j, an arbitrary compressed size, p_j, and a compression time q_j, such that $q_j = \frac{1}{d_i}$ for some $1 \leq i \leq z$, where $\{\frac{1}{d_i}\}$ is a divisible sequence. As we show in [1], even the more restricted case of *PRI*, with unit expanded size and uniform compression time $1/m$, is strongly NP-hard. Clearly, an approximation algorithm with an additive error of 1 is the best one can expect.

Algorithm $Alg_{div}(I, B)$ packs either $OPT - 1$ or $OPT - 2$ items, depending on several parameters of the instance (see below).

Property 1. For any item j, before j is placed, the packing consists of strips whose widths are multiples of q_j, such that the load along each strip is uniform.

Lemma 4. *Throughout Step (d), the gap between the loads on any two time-points in* $(0, 1]$ *is at most* $r_{j'}$.

$Alg_{div}(I, B)$:

1. Sort the items of I in non-decreasing order by weights, i.e., $w_1 \leq w_2 \leq \ldots$
2. Let I' be the longest prefix of the sorted list of total weight at most B.
3. Pack $|I'| - 1$ or $|I'| - 2$ items in the bin:
   (a) Remove from I' the item j' for which $r_{j'} = max_{j \in I'} r_j$.
   (b) If $w_{j'} < r_{j'}$ and $q_{j'} > min_{j \in I'} q_j$, remove from I' also an item with maximal weight.
   (c) Sort the remaining items in non-increasing order by compression times, i.e., $q_1 \geq q_2 \geq \ldots$. denote the sorted list by L.
   (d) while $L \neq \emptyset$, place the next item in the bin as follows:
      i. Let ℓ be the maximum load (=height) along $(0, 1]$ in the bin.
      ii. Let (x, ℓ) be the leftmost (with the minimal x-coordinate) point in the bin having load ℓ.
      iii. Pack j as compressed in time interval $(x, x + q_j]$, and as expanded in $(0, x]$, and $(x + q_j, 1]$.

Lemma 5. *The packing generated in Step (d) does not exceed the height* B.

Theorem 2. *If* $w_{j'} \leq r_{j'}$ *or* $q_{j'} = min_{j \in I'} q_j$, *then the algorithm packs* $OPT - 1$ *items. Otherwise, it packs* $OPT - 2$ *items.*

Proof. Since the total weight of items in OPT is at most B, and I' is the longest prefix of the sorted list having total weight at most B, it must be that $|I'| \geq OPT$. By Lemma 5, the algorithm packs all but one or two items from I', depending on the stated condition. ∎

Remark: For a given constant-size set U of items, it is possible to test in constant time whether all items of U can be packed. This can be done (also for arbitrary instances) by enumerating over all permutations of U and applying a greedy rule for each permutation (we omit the details). This implies that any algorithm which finds a solution in which $OPT/r - C$ are packed, can be converted to an $(r + \varepsilon)$-approximation algorithm. In particular, an algorithm that packs $OPT - \theta(1)$ items can be converted to a PTAS.

3.1 Uniform Compression Time

The above algorithm can be applied also if all items share the same compression time. That is, for all $j, q_j = q$ for any $0 < q < 1$. Note that we do not require q to be a unit fraction of the form $1/\left\lceil \frac{1}{q} \right\rceil$. Given q, let $\gamma = \left\lceil \frac{1}{q} \right\rceil$. We show that it is sufficient to consider only packings with a specific structure, in which the actual compression time of all items is exactly $1/\gamma$.

Lemma 6. *There exists an optimal packing in which the items are divided into* γ *groups, such that the items of group* $1 \leq i \leq \gamma$ *are all compressed during the interval* $(\frac{i-1}{\gamma}, \frac{i}{\gamma}]$.

Thus, for any instance with uniform q, it is possible to round down the compression times of all items to $1/\left\lceil \frac{1}{q} \right\rceil$, and apply $Alg_{div}(I, B)$ on the resulting instance - the compression times for a divisible sequence with a single element. Note that in this case, the condition in Step (b) does not hold (as $q_{j'} = min_{j \in I'} q_j$), and thus, the algorithm packs $OPT - 1$ item. Therefore,

Theorem 3. *Let I be an instance with uniform compression time. It is possible to pack $OPT(I) - 1$ items in polynomial time.*

Let I be an instance with uniform compression time $q = \frac{1}{\alpha}$, for some integer α. By applying $Alg_{div}(I, B)$ with a bin of capacity $B = w(I)$, all items except for one are packed. The following result is used in our algorithm for arbitrary instances (see Section 2).

Theorem 4. *Let I be an instance with uniform compression time $q = \frac{1}{\alpha}$. It is possible to pack $|I| - 1$ items in a bin of capacity $w(I)$.*

4 An AFPTAS for Instances with Uniform Size

In this section we present an improved approximation algorithm for instances with uniform expanded size. For such instances we assume, w.l.o.g., that for all items $s_j = 1$, and that the compression times of all items are positive (items with $q_j = 0$ can be added to the bin if it is not fully utilized by compressible items). We first describe the *Covering with Holes* technique that we use to derive our result.

4.1 Technique: Covering with Holes

Our approach is to utilize in the best way the *holes* created while items are packed as compressed. Each item j defines a hole h_j, which has a width $0 < q_j < 1$ and height $0 < p'_j < 1$, where $p'_j = 1 - p_j$. We describe formally the *Covering with Holes* problem, and its relation to *PRI*.

In the *Covering with Holes (CwH)* problem, we are given a number B, and a set H_I of n holes, such that each hole h_j is associated with a width $0 < q_j < 1$ and a height $0 < p'_j < 1$. The goal is to find the maximal h such that it is possible to cover an $h \times 1$ rectangle using at most $B + h$ holes. A solution for *CwH* is given by a set of holes $H = \{h_1, \cdots, h_\ell\}$ where $\ell \leq B + h$. For each hole h_j, the solution specifies what is the x-interval $X_j = (x_{1j}, x_{2j}]$ in which h_j is spanned, such that $x_{2j} - x_{1j} = q_j$. A solution covers an $h \times 1$ rectangle, if for every $0 \leq t \leq 1$ it holds that the total height of holes whose x-interval includes t is at least h.

Figure 2(a) presents a cover of a 1×1 rectangle with 7 holes. Note that the holes need not be placed in the covered area as rectangles. For example, hole h_4 spans along $(0.6, 1]$ and its height is 0.5. This hole corresponds to an item in the *PRI* instance having $q_j = 0.4$ and $p_j = 0.5$. Similarly, hole h_2 spans along $(0, 0.6]$ and its height is 0.3. This hole corresponds to a *PRI* item having $q_j = 0.6$ and

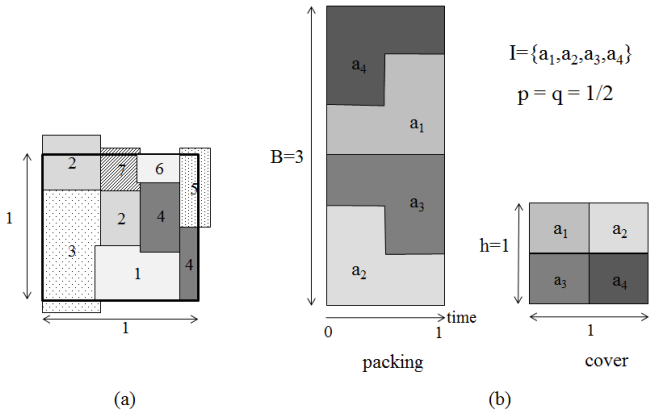


Fig. 2. (a) Covering a 1×1-rectangle with 7 holes. (b) A packing in a bin with capacity 3 and the corresponding cover of a 1×1 rectangle. The items define four holes of dimensions $\frac{1}{2} \times \frac{1}{2}$.

$p_j = 0.7$. Note also that it is possible to have overlapping holes as well as holes whose interval spans beyond the covered area.

We now show the relation between *CwH* and *PRI* (see Figure 2(b)).

Theorem 5. *Let I be an instance of* PRI *with uniform expanded size, and let H_I be the associated set of holes. It is possible to cover an $h \times 1$ rectangle using at most $B + h$ holes from H_I if and only if it is possible to pack $B + h$ items of I in a bin of capacity B.*

Theorem 5 holds for the maximal h such that at most $B+h$ holes cover an $h \times 1$ rectangle. Therefore, a solution for *CwH*, induces a solution for *PRI*.

Note that the above correspondence between the two problems holds for any instance of *PRI* with uniform expanded sizes.

4.2 Approximation Scheme

In the *uniform size* case, the expanded and the compressed size are uniform for all items and equal to 1 and p, respectively (for some $0 < p < 1 \leq B$). The compression times of the items (i.e., q_j), may be arbitrary.

Observation 2. *W.l.o.g., in any feasible packing, the number of compressed items is uniform along the interval* $(0, 1]$.

Recall that PRI is strongly NP-hard already for uniform size instances [1]. We use below our *covering with holes* technique to obtain an AFPTAS for such instances.

Assume the items are sorted such that $q_1 \geq q_2 \geq \ldots \geq q_n$. Clearly, if for two items j_1, j_2, it holds that $q_{j_i} > q_{j_2}$ then item j_1 can be accommodated in the space allocated for item j_2. Thus, w.l.o.g., an optimal solution packs the

first items in the instance. In covering terms, this means that the covering is performed using the widest holes.

Since the compressed size is uniform, all the holes determining the covering have height $(1-p)$. By Observation 2, in the *CwH* problem, there exists a cover in which the number of holes is uniform along the interval $(0,1]$. In other words, the holes are divided into $g = \left\lceil \frac{h}{1-p} \right\rceil$ groups (horizontal strips), such that group $1 \leq i \leq g$ covers the i-th strip of height $(1-p)$, of the rectangle. In other words, the covering problem in this case can be seen as a problem of maximizing the number of horizontal strips (each of height $(1-p)$), covered by rectangles of height $1-p$ and widths $q_1, q_2, \ldots$.

To obtain an approximate solution for *CwH*, we use an AFPTAS for the *bin covering* problem, defined as follows. Given is a set $\{a_1, \cdots, a_n\}$ of items, each item a_j has a size $s(a_j) \in (0,1)$. We need to pack the items into bins so as to maximize the number of bins that receive items of total size at least 1.

Let A_{bc} be such an AFPTAS, and let c be its asymptotic constant (such an AFPTAS with $c = 4$ is given in [11]). That is, given n items of sizes $s(1), \ldots, s(n)$, where $\forall i, s(i) < 1$, A_{bc} uses the items to cover b bins of size 1, where $b \geq (1-\varepsilon)b^* - c$, and b^* is the number of bins covered by an optimal solution. Given an instance I for *PRI*, let H_I be the corresponding instance for *CwH*. Define the following instance C_I for *bin covering*: for every hole $j \in H_I$ of size $(1-p) \times q_j$, include in C_I an item of size q_i.

Given H_I, h, we want to answer the following question: "Is it possible to cover an $h \times 1$ rectangle with at most $B + h$ holes from H_I?" Since, unless $P = NP$, this question is unlikely to be decided in polynomial-time, our algorithm answers a slightly different question. Specifically, *Decision Algorithm(H_I, h)* receives H_I and h as an input, and returns true if it is possible to cover a $((1-\varepsilon)h - c) \times 1$ rectangle with at most $(1-\varepsilon)(B+h) - c$ holes from H_I. The algorithm uses A_{bc} as a subroutine.

Decision Algorithm (H_I, h):

1. Let C_I be the input for *bin covering* corresponding to H_I.
2. Run A_{bc} on the first $(1-\varepsilon)(B+h) - c$ items in C_I (corresponding to the widest holes in H_I).
3. Let b be the number of bins covered by A_{bc}.
4. If $b \geq \frac{(1-\varepsilon)(h+B)-B-c}{1-p}$, return true else return false.

Lemma 7. *Let OPT_c be an optimal solution for* CwH *of the instance H_I. Assume that OPT_c covers a $(n^* - B) \times 1$ rectangle with at most n^* holes. Then, a $((1-\varepsilon)n^* - B - c) \times 1$ rectangle can be covered with at most $(1-\varepsilon)n^* - c$ holes and* Decision Algorithm(h) *returns true for $h = n^* - B$.*

To obtain an AFPTAS for PRI, we can use binary search to find the maximal $1 \leq h \leq n - B$ such that it is possible to cover a $((1-\varepsilon)h - c) \times 1$ rectangle with at most $(1-\varepsilon)(B+h) - c$ holes. We summarize in the next result.

Theorem 6. *Let n^* be the maximal number of items that can be packed in a bin of capacity B. Then the above scheme returns a packing of $(1-\varepsilon)n^* - c$ items, in a bin of capacity B.*

References

1. Albagli-Kim, S., Epstein, L., Shachnai, H., Tamir, T.: Packing Resizable Items with Application to Video Delivery over Wireless Networks. full version, http://www.cs.technion.ac.il/~hadas/PUB/PRI_full.pdf
2. Assman, S.B., Johnson, D.S., Kleitman, D.J., Leung, J.Y.T.: On a dual version of the one dimensional bin packing problem. J. Algorithms 5(4), 502–525 (1984)
3. The Cisco Visual Networking Index (VNI), http://www.cisco.com/en/US/netsol
4. Chaiken, S., Kleitman, D.J., Saks, M., Shearer, J.: Covering regions by rectangles. SIAM. J. on Algebraic and Discrete Methods 2(4), 394–410 (1981)
5. Chekuri, C., Khanna, S.: A PTAS for the multiple knapsack problem. In: Proc. of SODA, pp. 213–222 (2000)
6. Csirik, J., Frenk, J.B., Galambos, G., Rinnooy Kan, A.H.G.: Probabilistic analysis of algorithms for dual bin packing problems. J. Algorithms 12, 189–203 (1991)
7. Csirik, J., Johnson, D.S., Kenyon, C.: Better approximation algorithms for bin covering. In: Proceedings of SODA, pp. 557–566 (2001)
8. Fitzek, F.H.P., Hendrata, S., Seeling, P., Reisslein, M.: Video Streaming in Wireless Internet. CRC Press (2004)
9. Furht, B., Ahson, S.A. (eds.): Handbook of Mobile Broadcasting: DVB-H, DMB, ISDB-T, AND MEDIAFLO. Auerbach Publications (2008)
10. Franzblau, D.S., Kleitman, D.J.: An algorithm for covering polygons with rectangles. Information and Control Archive (1986)
11. Jansen, K., Solis-Oba, R.: An asymptotic fully polynomial time approximation scheme for bin covering. Theoretical Computer Science 306, 543–551 (2003)
12. Kellerer, H., Pferschy, U., Pisinger, D.: Knapsack Problems (2004)
13. Kenyon, C., Kenyon, R.: Tiling a Polygon with Rectangles. In: FOCS (1992)
14. Lee, C.C., Lee, D.T.: A simple on-line packing algorithm. J. of the ACM 32(3) (1985)
15. Pisinger, D.: Algorithms for Knapsack problems, Ph.D. Thesis, Dept. of Computer Science, University of Copenhagen, Denmark (February 1995)
16. Sahni, S.: Approximate algorithms for the 0/1 knapsack problem. J. of the ACM 22, 115–124 (1975)
17. Stoyan, Y., Romanova, T., Scheithauer, G., Krivulya, A.: Covering a polygonal region by rectangles. In: Computational Optimization and Applications (2009)
18. Tran, D.A., Nguyen, T.: Broadcasting techniques for video on demand in wireless networks. In: Handbook of Mobile Broadcasting. CRC Press (2008)

Symmetric Connectivity with Directional Antennas*

Rom Aschner[1], Matthew J. Katz[1], and Gila Morgenstern[2]

[1] Department of Computer Science, Ben-Gurion University, Israel
{romas,matya}@cs.bgu.ac.il
[2] Caesarea Rothschild Institute, University of Haifa, Israel
gilamor@cri.haifa.ac.il

Abstract. Let P be a set of points in the plane, representing transceivers equipped with a directional antenna of angle α and range r. The coverage area of the antenna at point p is a circular sector of angle α and radius r, whose orientation can be adjusted. For a given assignment of orientations, the induced *symmetric communication graph* (SCG) of P is the undirected graph, in which two vertices (i.e., points) u and v are connected by an edge if and only if v lies in u's sector and vice versa. In this paper we ask what is the smallest angle α for which there exists an integer $n = n(\alpha)$, such that for any set P of n antennas of angle α and unbounded range, one can orient the antennas so that (i) the induced SCG is connected, and (ii) the union of the corresponding wedges is the entire plane. We show (by construction) that the answer to this problem is $\alpha = \pi/2$, for which $n = 4$. Moreover, we prove that if Q_1 and Q_2 are two quadruplets of antennas of angle $\pi/2$ and unbounded range, separated by a line, to which one applies the above construction, independently, then the induced SCG of $Q_1 \cup Q_2$ is connected. This latter result enables us to apply the construction locally, and to solve the following two further problems.

In the first problem (*replacing omni-directional antennas with directional antennas*), we are given a connected unit disk graph, corresponding to a set P of omni-directional antennas of range 1, and the goal is to replace the omni-directional antennas by directional antennas of angle $\pi/2$ and range $r = O(1)$ and to orient them, such that the induced SCG is connected, and, moreover, is an $O(1)$-spanner of the unit disk graph, w.r.t. hop distance. In our solution $r = 14\sqrt{2}$ and the spanning ratio is 9. In the second problem (*orientation and power assignment*), we are given a set P of directional antennas of angle $\pi/2$ and adjustable range. The goal is to assign to each antenna p, an orientation and a range r_p, such that the resulting SCG is (i) connected, and (ii) $\sum_{p \in P} r_p^\beta$ is minimized, where $\beta \geq 1$ is a constant. For this problem, we devise an $O(1)$-approximation algorithm.

* Work by R. Aschner was partially supported by the Lynn and William Frankel Center for Computer Sciences. Work by M.J. Katz was partially supported by grant 1045/10 from the Israel Science Foundation, and by grant 2010074 from the United States – Israel Binational Science Foundation. Work by G. Morgenstern was partially supported by the Caesarea Rothschild Institute (CRI).

A. Bar-Noy and M.M. Halldórsson (Eds.): ALGOSENSORS 2012, LNCS 7718, pp. 18–29, 2013.

1 Introduction

Let P be a set of points in the plane, and assume that each point represents a transceiver equipped with a directional antenna. The *coverage area* of a directional antenna located at point p of angle α and range r, is a sector of angle α of the disk of radius r centered at p, where the orientation of the sector can be adjusted. We denote the coverage area of the antenna at p by W_p (since when assuming unbounded range the sector becomes a wedge). The induced *symmetric communication graph* (SCG) of P is the undirected graph over P, in which two vertices (i.e., points) u and v are connected by an edge if and only if $v \in W_u$ and $u \in W_v$.

The vast majority of the papers dealing with algorithmic problems motivated by wireless networks, consider omni-directional antennas, whose coverage area is often modeled by a disk. Many of these papers study problems, in which one has to assign radii (under some restrictions) to the underlying antennas so as to satisfy various coverage or communication requirements, while optimizing some measure, such as total power consumption. In the last few years, researches have begun to study such problems for directional antennas. Directional antennas have some noticeable advantages over omni-directional antennas. In particular, they require less energy to reach a point at a given distance, and they often facilitate in reducing the level of interferences in the network.

In this paper we ask the following question:

Problem 1. What is the smallest angle α for which there exists an integer $n = n(\alpha)$, such that for any set P of n points in the plane, representing transceivers equipped with directional antennas of angle α and unbounded range, one can orient the antennas so that (i) the induced SCG is connected, and (ii) the union of the corresponding wedges is the entire plane, i.e., for any point $x \in \mathbb{R}^2$, there exists a point $p \in P$, such that $x \in W_p$.

We would like to use the solution to this problem as a building block in the study of the following two important applications. These applications were studied under the asymmetric model of communication (where there is a directed edge from u to v if and only if $v \in W_u$), but not under the (more natural) symmetric model of communication, where they are considerably more difficult.

Replacing omni-directional antennas with directional antennas. Given a set P of points in the plane, let $UDG(P)$ be the *unit disk graph* of P (i.e., two points of P are connected by an edge if and only if the distance between them is at most 1), and assume that $UDG(P)$ is connected. Notice that $UDG(P)$ is the communication graph obtained by placing at each point of P an omni-directional antenna of range 1. The goal is to replace the omni-directional antennas with directional antennas of some small angle α and range r, such that (i) $r = O(1)$, (ii) the induced SCG is connected, and, moreover, (iii) the SCG is an $O(1)$-spanner of $UDG(P)$, w.r.t. hop distance (i.e., there exists a constant $t \geq 1$, such that, for each edge (p, q) of $UDG(P)$, there is a path between p and q in the SCG, consisting of at most t hops).

Orientation and power assignment. Given a set P of directional antennas of angle α and adjustable range, the goal is to assign to each antenna p, an orientation and a range r_p, such that the resulting SCG is connected, and $\sum_{p \in P} r_p^\beta$ is minimized, where $\beta \geq 1$ is the distance-power gradient (typically between 2 and 5).

Related work. A major challenge in the context of directional antennas is how to replace omni-directional antennas with directional antennas, such that (strong) connectivity is preserved, as well as other desirable properties, e.g., short range and similar hop distance. Several papers have considered this problem under the asymmetric model. Caragiannis et al. [7] consider the problem of orienting the directional antennas and fixing their range, such that the induced graph is strongly connected and the assigned range is minimized. They present a 3-approximation algorithm for any angle $\alpha \geq 0$; the maximum hop distance in their construction can be linear. In their survey chapter, Kranakis et al. [12] consider this problem in a more general setting, where each transceiver is equipped with $k \geq 1$ directional antennas. Damian and Flatland [10] show how to minimize both the range and the hop-ratio (w.r.t. the unit disk graph), for $\alpha \geq \pi/2$. Subsequently, Bose et al. [4] show how to do it for any $\alpha > 0$. Carmi et al. [8] were the first to study this problem under the symmetric model. They show that it is always possible to obtain a connected graph for $\alpha \geq \pi/3$, assuming the range is unbounded (i.e., equal to the diameter of the underlying point set). Later, a somewhat simpler construction was proposed by Ackerman et al. [1]. Carmi et al. [8] also observe that for $\alpha < \pi/3$ it is not always possible to orient the antennas such that the induced SCG is connected. Ben-Moshe et al. [3] investigate the problem of orienting quadrant antennas with only four possible orientations ($\pi/4$, $3\pi/4$, $5\pi/4$, and $7\pi/4$), and vertical half-strip antennas with only two possible orientations (up and down). Both problems are studied under the symmetric model.

The power assignment problem for omni-directional antennas is known to be NP-hard and was studied extensively; see, e.g., [5,6,9,11]. The orientation and power assignment problem, under the asymmetric model, was considered by Nijnatten [13], who observed that there exists a simple $O(1)$-approximation algorithm for any $\alpha \geq 0$. His solution is based on $O(1)$-approximation algorithms for the energy-efficient traveling salesman tour problem. The quality of his approximation does not depend on α. Notice that according to the observation of Carmi et al. [8] above, there does not always exist a solution to the orientation and power assignment problem under the symmetric model when $\alpha < \pi/3$.

Our results. In Section 2 we show that the solution to Problem 1 is $\alpha = \pi/2$, for which $n = 4$. Specifically, we show how to orient any four antennas of angle $\pi/2$, such that there is a path in the induced SCG between any two of them and they collectively cover the entire plane (assuming unbounded range). In order to use this construction as a building block in the solution of appropriate optimization problems, we need to be able to apply it locally, within a small geographic region, and to have a connection between nearby regions. Unfortunately, in Section 3 we give an example showing that we may not have such a connection.

We overcome this difficulty by proving the following theorem (which may be of independent interest). If Q_1 and Q_2 are two quadruplets of antennas of angle $\pi/2$ and unbounded range, separated by a line, to which one applies the above construction, independently, then the induced SCG of $Q_1 \cup Q_2$ is connected. In Section 4 we address the first application above. We show how to replace omni-directional antennas of range 1 with directional antennas of angle $\pi/2$ and range $14\sqrt{2}$ and orient them, such that the induced SCG is a 9-spanner of the unit disk graph, w.r.t. hop distance. In the full version of this paper (see [2]), we also study the second application. We show how to assign an orientation and range to each antenna in a given set of directional antennas of angle $\pi/2$, such that the induced SCG is connected and the total power consumption is at most some constant times the total power consumption in an optimal solution.

2 Connected Coverage of the Plane

In this section we consider Problem 1 What is the smallest angle α with the property that there exists a positive integer $n = n(\alpha)$, such that for any set P of at least n points, one can place directional antennas of angle α and unbounded range at the points of P, so that (i) the induced SCG is connected, and (ii) the plane is entirely covered by the antennas.

We show that the answer to the above question is $\alpha = \pi/2$. We first show that for $\alpha = \pi/2$, $n = 4$ is such an integer. Then, we show that for any $\alpha < \pi/2$, such an integer n does not exist.

Notation. We denote the antenna at point p by W_p, the left ray bounding W_p (when looking from p into W_p) by $\rho_p^{\nwarrow}$, and the right ray by $\rho_p^{\nearrow}$. The lines containing these two rays are denoted by $l(\rho_p^{\nwarrow})$ and $l(\rho_p^{\nearrow})$, respectively.

2.1 $\alpha = \pi/2$

Theorem 1. *Let P be a set of four points in the plane representing the locations of four transceivers equipped with directional antennas of angle $\pi/2$. Then, one can assign orientations to the antennas, such that the induced SCG is connected, and the plane is entirely covered by the four corresponding (unbounded) wedges.*

Proof. Denote the convex hull of P by $CH(P)$. We distinguish between the case where $CH(P)$ is a convex quadrilateral and the case where it is a triangle. If $CH(P)$ is a convex quadrilateral, then one of its angles is of size at most $\pi/2$. Each of the two diagonals of $CH(P)$ divides each of its corresponding two angles into two smaller angles, such that at least one of these smaller angles is of size at most $\pi/2$. Thus, at least 5 of the 8 angles defined by $CH(P)$ and its two diagonals are of size at most $\pi/2$. Denote the intersection point of the two diagonals by o. Then, there exist two adjacent vertices a, b of $CH(P)$, such that $\angle oab \leq \pi/2$ and $\angle oba \leq \pi/2$. Therefore, one can orient the antennas, such that the resulting SCG includes the two diagonals and the edge (a, b), and is thus connected.

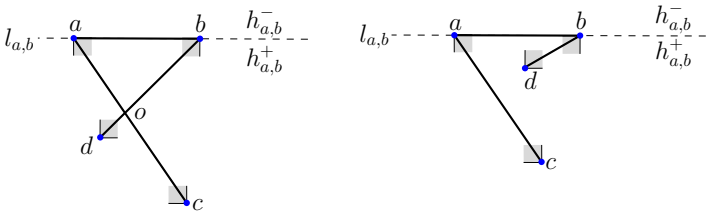


Fig. 1. Proof of Theorem 1

Let c and d be the other two vertices of $CH(P)$, such that c is adjacent to b. Then, we saw that one can orient the antennas, such that the resulting SCG includes the edges (a, c) and (b, d) and the edge (a, b). Denote by $l_{a,b}$ the line that passes through a and b, and by $h^+_{a,b}$ the closed half plane defined by $l_{a,b}$ and containing $CH(P)$. Reorient the antenna W_a at a (resp., W_b at b), such that it faces $CH(P)$ and one of its bounding rays passes through b (resp., a); see Figure 1(a). Notice that by doing so, we do not lose any of the graph edges, and, moreover, the half plane $h^+_{a,b}$ is entirely covered by the two antennas. To complete the proof we need to adjust the antennas at c and d, so that the half plane $h^-_{a,b}$ (on the other side of $l_{a,b}$) is also covered. Since c (resp. d) is covered by W_a (resp., W_b), we reorient W_c (resp., W_d) so that it is opposite W_a (resp., W_b); see Figure 1(a). By doing so, we do not lose any of the graph edges, and, moreover, the half plane $h^-_{a,b}$ is covered by $W_c \cup W_d$.

Assume now that $CH(P)$ is a triangle Δabc and that $d \in \Delta abc$. Then, Δabc has at least two angles of size at most $\pi/2$. W.l.o.g., assume $\angle cab \leq \pi/2$ and $\angle cba \leq \pi/2$. Orient W_a and W_b, as above, so that $h^+_{a,b}$ is covered by $W_a \cup W_b$. Notice that both W_a and W_b contain Δabc, and therefore both cover c and d. Orient W_c and W_d, so that $h^-_{a,b}$ is covered by $W_c \cup W_d$. The preceding observation implies that either W_c covers a and W_d covers b, or vice versa; see Figure 1(b). Thus, in any case, the obtained graph is connected.

Let us observe a few properties of the resulting structure. First, notice that the orientation of each antenna differs from the orientations of the other three by $\pi/2$, π, and $3\pi/2$, respectively. Second, each antenna is *coupled* with two of the others; namely, with those whose orientation differs from its own by $\pi/2$ and $3\pi/2$, respectively. For example, in Figure 1(a), W_c is coupled with W_b and with W_d. Notice that each such couple covers a half plane. E.g., W_c and W_b cover the appropriate half plane defined by $l(\rho^{\nearrow}_c)$, and W_c and W_d cover the appropriate half plane defined by $l(\rho^{\nearrow}_d)$.

2.2 $\alpha < \pi/2$

As observed by Carmi et al. [8], if $\alpha < \pi/3$, then it is not always possible to orient the antennas such that the resulting graph is connected. (Consider, for example, three antennas located at the vertices of an equilateral triangle.)

For $\alpha \geq \pi/3$, Carmi et al. [8], and subsequently Ackerman et al. [1], showed how to obtain a connected symmetric graph, for any set of antennas of angle α. However, their construction does not ensure that the union of the wedges covers the entire plane. Actually, it is not always possible to orient a set of antennas with angle $\alpha < \pi/2$, so that the induced graph is connected and the entire plane is covered. This is true even in the asymmetric model where one only requires strong connectivity, as observed by Bose et al. [4]. To see this, consider, e.g., a set P of points on a vertical line segment s. In order to cover a point that lies, e.g., far enough to the right of s, at least one of the corresponding antennas must be oriented such that its wedge is empty of points of P (except for the point at its apex). This antenna is isolated in the resulting SCG.

3 Separated Quadruplets Are Connected

Let A and B be two quadruplets of points (representing transceivers) in the plane, and assume that each of the transceivers is equipped with a directional antenna of angle $\pi/2$. Orient the antennas corresponding to the points in A (resp., in B), such that they satisfy the conditions of Theorem 1. Clearly, each point in A is covered by at least one antenna corresponding to a point in B, and vice versa. Unfortunately, this does not imply that the SCG induced by $A \cup B$ is connected; see Figure 2 for an example where there is no edge between A and B in the SCG of $A \cup B$. Theorem 2 below is crucial for our subsequent applications. It states that if the quadruplets A and B can be separated by a line, then the induced SCG is surely connected.

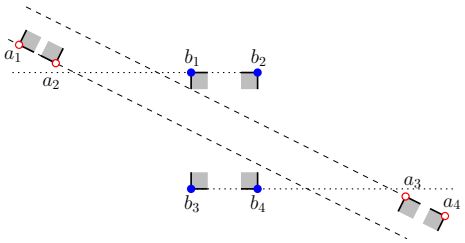


Fig. 2. The induced SCG is not connected

Theorem 2. *Let A, B be two sets of four points each, and let the (antennas corresponding to the) points of A and, independently, the points of B be oriented as in the proof of Theorem 1. If there exists a line l that separates between A and B, then the SCG induced by $A \cup B$ is connected.*

Proof. It is enough to show that there exist a point $a \in A$ and a point $b \in B$ that cover each other (i.e., $a \in W_b$ and $b \in W_a$). Assume w.l.o.g. that l is vertical and that the points of A (resp., B) lie to the left (resp., right) of l. Denote by h_A

(resp., h_B) the half plane that is defined by l and contains A (resp., B). Let m_A be the smallest number, such that one can pick m_A points of A that (together) cover h_B. Clearly, m_A is either 2 or 3. We distinguish between two cases. In the first case, at least one of the two numbers m_A and m_B is 2, where m_B is defined analogously. In the second case, both m_A and m_B are 3.

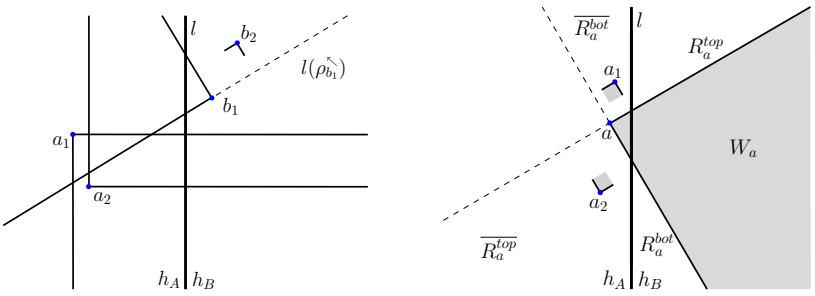


Fig. 3. Proof of Theorem 2 — Cases 1 and 2

Case 1. There exist two points from one set that (together) cover the half plane containing the other set. W.l.o.g., assume points $a_1, a_2 \in A$ cover h_B, and a_1 is not below a_2. In this case, a_1 and a_2 form a couple (see above), and the half plane covered by them contains h_B. That is, the rays $\rho_{a_1}^{\nearrow}$ and $\rho_{a_2}^{\nwarrow}$ are parallel to l, where $\rho_{a_1}^{\nearrow}$ is pointing downwards and $\rho_{a_2}^{\nwarrow}$ is pointing upwards, and $\rho_{a_1}^{\nwarrow}$ and $\rho_{a_2}^{\nearrow}$ are perpendicular to l and pointing rightwards; see Figure 3(a). Let $b_1 \in B$ be a point that covers the point a_1. If b_1 lies in W_{a_1}, then we are done, since a_1 and b_1 cover each other. Assume, therefore, that b_1 lies in $h_B \setminus W_{a_1} \subseteq W_{a_2}$, and that b_1 does not cover a_2 (since, otherwise, a_2 and b_1 cover each other and we are done). It follows that $l(\rho_{b_1}^{\nwarrow})$ crosses the line segment $\overline{a_1 a_2}$. Let b_2 be the point in B, such that b_1 and b_2 form a couple, and b_2's orientation is equal to b_1's orientation plus $\pi/2$ (adding counterclockwise). Then, b_2 lies above (or on) $l(\rho_{b_1}^{\nwarrow})$ (and, of course, $b_2 \in h_B$), and a_2 lies in W_{b_2}. But, the former assertion implies that b_2 lies in W_{a_2}, hence a_2 and b_2 cover each other, and we are done.

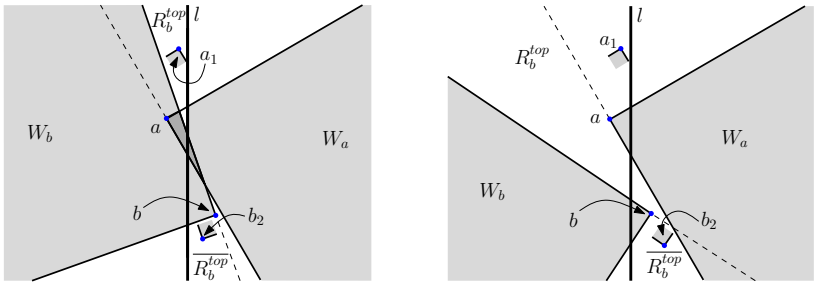
Case 2. h_B is covered by three points of A (but not by two), and h_A is covered by three points of B (but not by two). Notice that on each side of l there exists a point, whose wedge divides the half plane on the other side of l into three disjoint regions. More precisely, there exists $a \in A$ that divides h_B into three disjoint regions: $W_a \cap h_B$, R_a^{top}, and R_a^{bot}; see Figure 3(b). We denote the "complement" regions of R_a^{top} and R_a^{bot} by $\overline{R_a^{top}}$ and $\overline{R_a^{bot}}$, respectively, where $\overline{R_a^{top}}$ (resp., $\overline{R_a^{bot}}$) is obtained by rotating R_a^{top} (resp., R_a^{bot}) around its apex by π. Notice that $\overline{R_a^{top}} \cap \overline{R_a^{bot}} \neq \emptyset$. Let a_1 be the point in A, such that a and a_1 form a couple, and a_1's orientation is equal to a's orientation minus $\pi/2$. Then, a_1 lies above $l(\rho_a^{\nearrow})$ (i.e., $a_1 \in \overline{R_a^{bot}}$) and a_1 covers R_a^{bot}. Similarly, let a_2 be the point in A, such that

a and a_2 form a couple, and a_2's orientation is equal to a's orientation plus $\pi/2$. Then, $a_2 \in \overline{R_a^{top}}$ and a_2 covers R_a^{top}; see Figure 3(b) for an illustration.

As above, let b be a point in B that divides h_A into three disjoint regions, $W_b \cap h_A$, R_b^{top}, and R_b^{bot}; denote the "complement" regions of R_b^{top} and R_b^{bot} by $\overline{R_b^{top}}$ and $\overline{R_b^{bot}}$, respectively; let $b_1, b_2 \in B$, $b_1 \in \overline{R_b^{bot}}$ and $b_2 \in \overline{R_b^{top}}$, such that b_1 covers R_b^{bot} and b_2 covers R_b^{top}.

We now show that there exist two points, one in A and one in B, that cover each other. We distinguish between a few subcases.

Case 2a. $b \in W_a$ and $a \in W_b$. That is, a and b cover each other.



(a) Case 2b (b) Case 2c

Fig. 4. Proof of Theorem 2 — Cases 2b and 2c

Case 2b. $b \in R_a^{bot}$ (that is, b is covered by a_1) and $a \in W_b$; see Figure 4(a). Assume that $a_1 \notin W_b$ (otherwise, a_1 and b cover each other), then, $a_1 \in R_b^{top}$ and thus covers $\overline{R_b^{top}}$. Recall that $b_2 \in \overline{R_b^{top}}$ and covers R_b^{top}. We conclude that a_1 and b_2 cover each other.

Case 2c. $b \in R_a^{bot}$ and $a \notin W_b$; see Figure 4(b). Then, $a, a_1 \in R_b^{top}$, and, therefore, b_2 covers both a and a_1. Now, if b_2 lies in W_a, then a and b_2 cover each other, and we are done. Otherwise, $b_2 \in R_a^{bot}$, but then, b_2 and a_1 cover each other.

Finally, it is easy to verify that all other subcases are symmetric to either Case 2b or Case 2c.

4 Replacing Omni-Directional Antennas with Directional Antennas

Let P be a set of n points in the plane. The *unit disk graph* of P, denoted $UDG(P)$, is the graph over P, in which there is an edge between two points if and only if the distance between them is at most 1. Notice that $UDG(P)$ is the communication graph obtained, when each point of P represents a transceiver with an omni-directional antenna of range 1. Assume that $UDG(P)$ is connected.

Our goal in this section is to replace the omni-directional antennas with directional antennas of angle $\pi/2$ and range $r = O(1)$ and to orient them, such that the induced *symmetric* communication graph is (i) connected, and (ii) a c-spanner of $UDG(P)$, with respect to hop distance, where c is an appropriate constant. We show that this can be done for $r = 14\sqrt{2}$ and $c = 9$.

The main idea underlying our construction is to apply Theorem 1 multiple times, each time to a cluster of points within a small region, and to use Theorem 2 to establish that the SCG induced by any two such clusters is connected (assuming unbounded range).

We now describe our construction. Lay a regular grid $\mathcal{G}$ over P, such that the length of a cell side is 7. (A cell $\mathcal{C}$ with bottom-left corner (x_0, y_0) is the semi-open square $[x_0, x_0 + 7) \times [y_0, y_0 + 7)$.) For a cell $\mathcal{C}$ of $\mathcal{G}$, the *block* of $\mathcal{C}$ is the 3×3 portion of $\mathcal{G}$ centered at $\mathcal{C}$. Each of the 8 cells surrounding $\mathcal{C}$ is a *neighbor* of $\mathcal{C}$. A cell of $\mathcal{G}$ is considered *full* if it contains at least four points of P. It is considered *non-full* if it contains at least one and at most three points of P. Proposition 1 below is analogous to a proposition of Bose et al. [4], referring to a similar grid.

Proposition 1. ([4]) *Let $\mathcal{C}$ be a cell of $\mathcal{G}$. Then, any path in $UDG(P)$ that begins at a point in $\mathcal{C}$ and exits the block of $\mathcal{C}$, must pass through a full cell in $\mathcal{C}$'s block (not including $\mathcal{C}$ itself, which may or may not be full). In particular, if there are points of P outside $\mathcal{C}$'s block, then at least one of $\mathcal{C}$'s neighbors is full.*

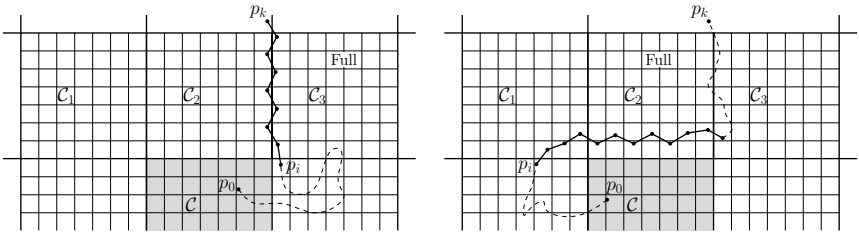


Fig. 5. Proposition 1

Proof. Let $\Pi = \langle p_0, p_1, \cdots, p_k \rangle$ be a path that begins at a point $p_0 \in \mathcal{C}$ and exits $\mathcal{C}$'s block, where p_k is the first point in Π that is not in $\mathcal{C}$'s block. Assume, e.g., that p_k lies on or above the line containing the top side of $\mathcal{C}$'s block, and let p_i be the last point in Π that lies below the line containing the top side of $\mathcal{C}$; see Figure 5. Then, all points in the subpath $\langle p_i, p_{i+1}, \cdots, p_k \rangle$ of Π, except for the two extreme ones, lie in the union of the top three cells of $\mathcal{C}$'s block, and their number (excluding the two extreme points) is at least 7. Let $\mathcal{C}_1$, $\mathcal{C}_2$, and $\mathcal{C}_3$ be the top three cells, from left to right, of $\mathcal{C}$'s block. If $\{p_{i+1}, \ldots, p_{k-1}\}$ is contained in only one of these cells, or, alternatively, in only two of these cells, then at least one of these cells contains at least 4 points, and we are done (Figure 5(a)). Otherwise, at least one of the points lies in $\mathcal{C}_1$ and at least one of the points lies in $\mathcal{C}_3$, implying that $\mathcal{C}_2$ contains at least 7 points (Figure 5(b)).

If none of the grid cells is full, then, it is easy to see that P is contained in a 14×14 square. In this case, we can apply Theorem 1 to an arbitrary subset P' of four points of P, and orient each of the other points of P towards a point of P' that covers it. By setting $r = 14\sqrt{2}$, we obtain a SCG, in which the hop distance between any two points is at most 5 (see below). We thus may assume that at least one of the grid cells is full.

For each cell $\mathcal{C}$ of $\mathcal{G}$, we orient the points in $\mathcal{C}$ as follows. If $\mathcal{C}$ is full, then arbitrarily pick four points in $\mathcal{C}$ as $\mathcal{C}$'s *hub points*, and orient these hub points according to Theorem 1. Next, orient each non-hub point in $\mathcal{C}$ towards one of the hub points of $\mathcal{C}$ that covers it. If $\mathcal{C}$ is non-full, then, for each point $p \in \mathcal{C}$, pick a full cell closest to p, that is, a full cell containing a point x_p, such that the hop distance (in $UDG(P)$) from p to x_p is not greater than the hop distance from p to any other point of P lying in a full cell. Denote this cell by $\mathcal{C}_{F(p)}$. Notice that by Proposition 1, $\mathcal{C}_{F(p)}$ is a neighbor of $\mathcal{C}$. Orient each $p \in \mathcal{C}$ towards a hub point of $\mathcal{C}_{F(p)}$ that covers it. Finally, set $r = 14\sqrt{2}$, so that each antenna can reach any point in its own cell or in a neighboring cell, provided that this point is within the antenna's wedge.

Let G be the resulting SCG. Lemma 2, below, states that G is a 9-spanner of $UDG(P)$, w.r.t. hop distance. In particular, since $UDG(P)$ is connected, then so is G. We first prove the following auxiliary lemma.

Lemma 1. *Let (p, q) be an edge of $UDG(P)$, and let $\mathcal{C}_p$ and $\mathcal{C}_q$ be the cells of $\mathcal{G}$, such that $p \in \mathcal{C}_p$ and $q \in \mathcal{C}_q$.*

(i) If $\mathcal{C}_p$ and $\mathcal{C}_q$ are both full, then either $\mathcal{C}_p = \mathcal{C}_q$ or $\mathcal{C}_p$ and $\mathcal{C}_q$ are neighbors.
(ii) If $\mathcal{C}_p$ is full and $\mathcal{C}_q$ is non-full, then either $\mathcal{C}_p = \mathcal{C}_{F(q)}$ or $\mathcal{C}_p$ and $\mathcal{C}_{F(q)}$ are neighbors.
(iii) If $\mathcal{C}_p$ and $\mathcal{C}_q$ are both non-full, then either $\mathcal{C}_{F(p)} = \mathcal{C}_{F(q)}$ or $\mathcal{C}_{F(p)}$ and $\mathcal{C}_{F(q)}$ are neighbors.

Proof. (i) This is obvious, since the Euclidean distance between p and q is at most 1 and the side length of a cell is 7.
(ii) Let $x_q \in \mathcal{C}_{F(q)}$ be a point that determines the hop distance between q and $\mathcal{C}_{F(q)}$ (see above). By construction, the hop distance in $UDG(P)$ from q to x_q is not greater than the hop distance from q to p, which is 1. Thus, the Euclidean distance between $p \in \mathcal{C}_p$ and $x_q \in \mathcal{C}_{F(q)}$ is at most 2, whereas the side length of a cell is 7. We conclude therefore that either $\mathcal{C}_p = \mathcal{C}_{F(q)}$ or $\mathcal{C}_p$ and $\mathcal{C}_{F(q)}$ are neighbors.
(iii) Assume to the contrary that $\mathcal{C}_{F(p)} \neq \mathcal{C}_{F(q)}$ and that $\mathcal{C}_{F(p)}$ and $\mathcal{C}_{F(q)}$ are not neighbors. Let $x_p \in \mathcal{C}_{F(p)}$ (resp., $x_q \in \mathcal{C}_{F(q)}$) be a point that determines the hop distance between p and $\mathcal{C}_{F(p)}$ (resp., q and $\mathcal{C}_{F(q)}$). Consider the path Π in $UDG(P)$ that is obtained by concatenating the shortest path from x_p to p, the edge (p, q), and the shortest path from q to x_q. By our assumption, $\mathcal{C}_{F(q)}$ is not in $\mathcal{C}_{F(p)}$'s block (and vice versa), thus Π starts at $\mathcal{C}_{F(p)}$ and exits its block. By Proposition 1, Π passes through a full cell $\mathcal{C}$ in $\mathcal{C}_{F(p)}$'s block, other than $\mathcal{C}_{F(p)}$ (and other than $\mathcal{C}_{F(q)}$, which is not in $\mathcal{C}_{F(p)}$'s block). Since $\mathcal{C}$ is full, $\mathcal{C} \neq \mathcal{C}_p, \mathcal{C}_q$. Now, if Π visits $\mathcal{C}$ before it visits the point p, then we get that $\mathcal{C}$ is a full cell

closer to p than is $\mathcal{C}_{F(p)}$, and, if Π visits $\mathcal{C}$ after it visits q, then we get that $\mathcal{C}$ is a full cell closer to q than is $\mathcal{C}_{F(q)}$. Thus, in both cases we arrive at a contradiction.

Lemma 2. *G is a 9-spanner of $UDG(P)$, w.r.t. hop distance.*

Proof. Let (p, q) be an edge of $UDG(P)$. We show that G contains a path from p to q consisting of at most 9 edges. Let $\mathcal{C}_p$ and $\mathcal{C}_q$ be the cells of $\mathcal{G}$, such that $p \in \mathcal{C}_p$ and $q \in \mathcal{C}_q$. We distinguish between the three cases listed in Lemma 1.

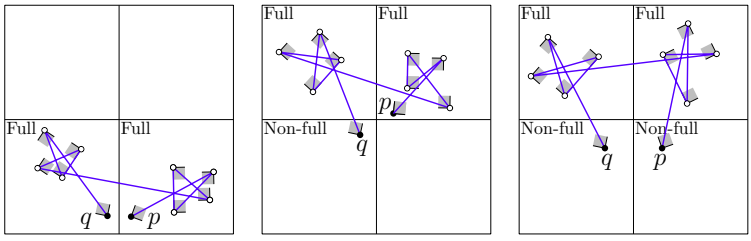


Fig. 6. G is a 9-spanner of $UDG(P)$, w.r.t. hop distance

(i) Consider first the case where $\mathcal{C}_p$ and $\mathcal{C}_q$ are both full. Then, by Lemma 1, either $\mathcal{C}_p = \mathcal{C}_q$ or $\mathcal{C}_p$ and $\mathcal{C}_q$ are neighbors. Notice that p is either a hub point of $\mathcal{C}_p$, or it is connected to one by a single edge; and the same holds for q and $\mathcal{C}_q$. Assume first that $\mathcal{C}_p = \mathcal{C}_q$, then, there exists a path in G from p to q consisting of at most 5 edges. Indeed, such a path starts at p, passes through at most four hub points of $\mathcal{C}_p$, and ends at q. Assume now that $\mathcal{C}_p$ and $\mathcal{C}_q$ are neighbors. By Theorem 2 and since the range of each antenna is sufficient to reach any point in any neighboring cell, there exists an edge in G connecting a hub point of $\mathcal{C}_p$ with a hub point of $\mathcal{C}_q$. Consequently, G contains a path from p to q consisting of at most 9 edges. Indeed, such a path starts at p, passes through at most four hub points of $\mathcal{C}_p$, continues to a hub point of $\mathcal{C}_q$, passes through at most four hub points of $\mathcal{C}_q$, and finally ends at q; see Figure 6(a) for an illustration.

(ii) Consider now the case where one of the cells, say $\mathcal{C}_p$, is full and the other is non-full. By construction, q is oriented to a hub point of a neighboring full cell $\mathcal{C}_{F(q)}$ that covers it. By Lemma 1, either $\mathcal{C}_p = \mathcal{C}_{F(q)}$ or $\mathcal{C}_p$ and $\mathcal{C}_{F(q)}$ are neighbors. Therefore, as in case (i) above, G contains a path from p to q consisting of at most 9 edges; see Figure 6(b) for an illustration.

(iii) Finally, consider the case where $\mathcal{C}_p$ and $\mathcal{C}_q$ are both non-full. By construction, p (resp., q) is connected by an edge to a hub point of a full cell $\mathcal{C}_{F(p)}$ (resp., $\mathcal{C}_{F(q)}$) that covers it. By Lemma 1, either $\mathcal{C}_{F(p)} = \mathcal{C}_{F(q)}$ or $\mathcal{C}_{F(p)}$ and $\mathcal{C}_{F(q)}$ are neighbors. Therefore, as in case (i) above, G contains a path from p to q consisting of at most 9 edges; see Figure 6(c) for an illustration.

Theorem 3 summarizes the main result of this section.

Theorem 3. *Let P be a set of points, where each point represents a transceiver equipped with an omni-directional antenna of range 1, and assume that $UDG(P)$ is connected. Then, one can replace the omni-directional antennas with directional antennas of angle $\pi/2$ and range $14\sqrt{2}$, such that the induced SCG is (i) connected, and (ii) a 9-spanner of $UDG(P)$, w.r.t. hop distance.*

Remark. In the full version of this paper (see [2]), we show how to reduce the hop ratio from 9 to 8, by picking the hub points (in each full cell) more carefully.

References

1. Ackerman, E., Gelander, T., Pinchasi, R.: Ice-creams and wedge graphs. CoRR abs/1106.0855 (2011)
2. Aschner, R., Katz, M.J., Morgenstern, G.: Symmetric connectivity with directional antennas. CoRR abs/1108.0492 (2011)
3. Ben-Moshe, B., Carmi, P., Chaitman, L., Katz, M.J., Morgenstern, G., Stein, Y.: Direction assignment in wireless networks. In: Proc. 22nd Canadian Conf. on Computational Geometry, pp. 39–42 (2010)
4. Bose, P., Carmi, P., Damian, M., Flatland, R., Katz, M.J., Maheshwari, A.: Switching to Directional Antennas with Constant Increase in Radius and Hop Distance. In: Dehne, F., Iacono, J., Sack, J.-R. (eds.) WADS 2011. LNCS, vol. 6844, pp. 134–146. Springer, Heidelberg (2011)
5. Calinescu, G.: Min-power strong connectivity. In: Proc. 13th Internat. Workshop on Approximation Algorithms for Combinatorial Optimization Problems, pp. 67–80 (2010)
6. Calinescu, G., Mandoiu, I.I., Zelikovsky, A.: Symmetric connectivity with minimum power consumption in radio networks. In: Proc. 2nd IFIP Internat. Conf. on Theoretical Computer Science, pp. 119–130 (2002)
7. Caragiannis, I., Kaklamanis, C., Kranakis, E., Krizanc, D., Wiese, A.: Communication in wireless networks with directional antennas. In: Proc. 20th ACM Sympos. on Parallelism in Algorithms and Architectures, pp. 344–351 (2008)
8. Carmi, P., Katz, M.J., Lotker, Z., Rosén, A.: Connectivity guarantees for wireless networks with directional antennas. Computational Geometry: Theory and Applications 44(9), 477–485 (2011)
9. Clementi, A.E.F., Penna, P., Silvestri, R.: Hardness Results for the Power Range Assignment Problem in Packet Radio Networks. In: Hochbaum, D.S., Jansen, K., Rolim, J.D.P., Sinclair, A. (eds.) RANDOM-APPROX 1999. LNCS, vol. 1671, pp. 197–208. Springer, Heidelberg (1999)
10. Damian, M., Flatland, R.: Connectivity of graphs induced by directional antennas. CoRR abs/1008.3889 (2010)
11. Kirousis, L.M., Kranakis, E., Krizanc, D., Pelc, A.: Power consumption in packet radio networks. Theoretical Computer Science 243, 289–305 (2000)
12. Kranakis, E., Krizanc, D., Morales, O.: Maintaining connectivity in sensor networks using directional antennae. In: Nikoletseas, S., Rolim, J.D.P. (eds.) Theoretical Aspects of Distributed Computing in Sensor Networks, ch. 3, pp. 83–110. Springer
13. van Nijnatten, F.: Range Assignment with Directional Antennas. Master's Thesis, Technische Universiteit Eindhoven (2008)

Comparative Study of Approximation Algorithms and Heuristics for SINR Scheduling with Power Control*

Lukas Belke, Thomas Kesselheim[1],
Arie M.C.A. Koster[2], and Berthold Vöcking[1]

[1] Department of Computer Science, RWTH Aachen University, Germany
[2] Lehrstuhl II für Mathematik, RWTH Aachen University, Germany

Abstract. Various recent theoretical studies have achieved considerable progress in understanding combined link scheduling and power control in wireless networks with SINR constraints. These analyses were mainly focused on designing and analyzing approximation algorithms with provable approximation guarantees. While these studies revealed interesting effects from a theoretical perspective, so far there has not been a systematic evaluation of the theoretical results in simulations. In this paper, we examine the performance of various approximation algorithms and heuristics for the common scheduling problems on instances generated by different random network models, e.g., taking clustering effects into account. Using non-convex optimization, we are able to compute the theoretical optima for some of these instances such that the performance of the different algorithms can be compared with these optima.

The simulations support the practical relevance of the theoretical findings. For example, setting transmission power by a square-root power assignment, the network's capacity increases significantly in comparison to uniform power assignments. Furthermore, the developed approximation algorithms are able to exploit this gap providing in general a better performance than any algorithm using uniform transmission powers, even with unlimited computational power. The obtained results are robust against changes in parameters and network generation models.

1 Introduction

Triggered by a seminal work by Moscibroda and Wattenhofer [15], recent algorithmic research on wireless networks has mostly been considering models based on the signal-to-interference-plus-noise ratio (SINR). Using such models, important aspects such as aggregation of interference or different transmission powers are taken into account. Most prominently, approximation algorithms for the combined problem of link scheduling and power control (see Section 1.2) have been considered. However, devising approximation algorithms in a worst-case model

* This work has been supported by the UMIC Research Centre, RWTH Aachen University.

A. Bar-Noy and M.M. Halldórsson (Eds.): ALGOSENSORS 2012, LNCS 7718, pp. 30–41, 2013.

does not necessarily lead to practical results since the approximation ratio might attain its maximum only in artificially created instances. There has not been a systematic evaluation of the theoretical results in simulations up to now. In this paper, we examine how approximation algorithms that were originally designed in the context of theoretical worst-case analyses perform on randomly generated instances.

We focus on the *capacity-maximization problem*. Given n communication links, each being a pair of a sender and a receiver node, the task is to select a maximum subset of them and assign transmission powers such that the SINR is above some threshold at the receiver of each link in the set. Already Moscibroda and Wattenhofer showed that setting transmission powers appropriately is inevitable for good worst-case performance guarantees in certain networks. For example, for the capacity-maximization problem there are networks in which all n links can be scheduled with the right transmission powers. Restricting all transmissions to use uniform powers even the best solution is hardly better than the trivial one that only selects a single link.

We strive to examine whether these results are particular to the respective worst-case networks or whether similar results can still be observed in randomly generated networks. We tackle these questions from two sides. In the first step, we investigate whether the effect of power control is as important as suggested by theoretical studies. For this purpose, we compare the theoretical optima of several power assignments, including the respective optimal choice. As the involved problems are NP hard, computing each of these optima needs exponential running times. For this reason, in the second step, we focus on approximation algorithms and heuristics that run in polynomial time. We compare the computed approximate solutions with the respective optimum. In both steps, we use non-convex optimization to compute the optimal solution.

1.1 Formal Problem Statement and the SINR Model

We model the interference constraints as follows (cf. [8]). We assume that the network nodes are located in a metric space. In our simulations this is the plane with the Euclidean distance. We model the signal propagation as follows. Assume that some sender s transmits a signal at power level p, then receiver r receives this signal at a strength of $p/d(s,r)^\alpha$. Here, $\alpha > 0$ denotes the path-loss exponent, that is typically assumed to be between 2 and 6. The transmission can be successfully received if its signal-to-interference-plus-noise ratio (SINR) is above some threshold $\beta > 0$. That is, the strength of the intended signal has to be β-times as strong as all simultaneous interfering signals plus ambient noise.

Formally, a set S of sender-receiver pairs (links) is feasible under a power assignment $p\colon S \to \mathbb{R}_{\geq 0}$ if for all $\ell = (s,r) \in S$ the SINR constraint

$$\frac{p(\ell)}{d(s,r)^\alpha} \geq \beta \left(\sum_{\ell'=(s',r')\in S, \ell' \neq \ell} \frac{p(\ell')}{d(s',r)^\alpha} + N \right)$$

is fulfilled, where $N \geq 0$ denotes ambient noise. In the *capacity-maximization problem*, one is given a set $\mathcal{R}$ of n links. The task is to select a subset $S \subseteq \mathcal{R}$ and

a power assignment $p\colon S \to \mathbb{R}_{\geq 0}$ such that S is feasible under p. The objective is to maximize $|S|$.

1.2 Considered Approximation Algorithms

For the problem of combined link scheduling and power control a number of heuristic approaches have been proposed to solve this problem exactly [3] or approximately [18,4]. These algorithms, however, rely on stochastic assumptions on the distribution of the network nodes. They do not run in polynomial time or do not provide provable performance guarantees [14].

First theoretical studies of approximation algorithms for link scheduling in the SINR model concentrated on uniform power assignments. Goussevskaia et al. [7] presented an approximation algorithm for the capacity maximization problem that achieves a constant approximation factor with respect to the optimal solution using uniform power assignments. That is, the computed solution is compared to only the ones using the same power for each transmission. A simplified version of this algorithm has been presented by Halldórsson and Wattenhofer [11], which has been implemented for the simulations.

Andrews and Dinitz [1] also gave an approximation algorithm for capacity maximization with uniform transmission powers. Their analysis, however, is with respect to an arbitrary power assignment. The approximation factor is shown to be $O(\log \Delta)$, where Δ is the ratio between the largest and the smallest distance between a sender a receiver. Furthermore, they proved the combined scheduling and power control problem to be NP hard.

Going beyond uniform power assignments, oblivious power assignments have been considered. Here, the transmission power may only depend on the distance between the sender and the respective receiver. The most prominent representatives are linear and square-root power assignments. In a linear power assignment $p(\ell)$ is proportional to $d(\ell)^\alpha$ for all links ℓ. The obtained theoretical results are comparable to the ones with uniform power assignments [5]. Better ones can be achieved using square-root (or mean) power assignments, which set all powers $p(\ell)$ proportional to $\sqrt{d(\ell)^\alpha}$. They have been introduced by Fanghänel et al. [5], who showed that they allow to achieve $\log^{O(1)} n$ approximations for capacity maximization in a bidirectional model. Halldórsson [9] extended this study to the standard directed communication model, giving an $O(\log\log\Delta \cdot \log n)$ approximation algorithm for capacity maximization. Halldórsson and Mitra [10] improved this factor to $O(\log\log\Delta + \log n)$. Up to now, this is the best approximation algorithm using square-root power assignments. Therefore, it has been implemented for our simulations as well.

All oblivious power assignments, however, have the drawback that the approximation factors do and have to depend on Δ. Fanghänel et al. [5] showed that for any oblivious power assignment there are certain network instances in which the optimal solution with arbitrary power assignments is of size $\Omega(n)$. Restricting the transmission powers to the respective oblivious power assignment the optimum degrades to $O(1)$. That means that an algorithm using uniform transmission powers cannot achieve a better worst-case approximation factor than

$O(n)$, which is also achieved by the trivial algorithm that just selects a single link. The first approximation algorithm guaranteeing an approximation factor that is independent of the network was presented by Kesselheim [12]. An extension shows how to deal with restricted transmission powers [16]. We consider the original algorithm for our studies.

1.3 Our Results

One of the most striking results from the design of approximation algorithms is the square-root power assignment. Theoretical studies show that it is superior to uniform and linear power assignments but optimizing the power assignment for the respective instance can still yield better results. Our simulation results support this insight. Using non-convex optimization, we compute the optimal subset of links with respect to arbitrary power assignments and also with respect to uniform, linear, and square-root power assignments. The results indicate that the square-root power assignment is able to partly exploit the potential of power control. That is, the optimal solution is significantly better than the ones using uniform or linear power assignments. However, like in theoretical worst-case considerations, also in our randomly generated instances setting transmission powers yields better results. We carry out these simulations for networks of up to 800 links.

The mentioned computation of the optimal solution is NP hard. Therefore, the computation needs exponential time in general. Also in our simulations, running times get prohibitively large in large instances. For this reason, approximation algorithms have been developed that need a linear or quadratic running time in the number of links. We execute these algorithms on randomly generated instances, consisting of up to 1600 links. These simulations also support the theoretical findings. Algorithms explicitly setting the transmission powers are able to outperform the optimum with uniform power assignments. That is, using power control these algorithms are able to provide better solutions than any algorithm using uniform transmission powers could compute, even with unlimited computational power. However, the theoretical inferiority of the square-root power assignment compared to the constant-factor approximation [12] cannot be observed in the randomly generated instances. Interestingly, all approximation algorithms and heuristics achieve comparable results.

In order to eliminate effects of particular networks or parameters, we repeat all simulations with differently generated networks, with and without clustering. Furthermore, we apply multiple parameter settings. This changes the absolute number of links that can be scheduled. In either case, the relative performance is similar. In particular, the ranking of the algorithms and power assignments is consistent.

2 Generation of Network Instances

For our simulations, we construct network instances applying two different techniques. In each case, we randomly place n sender-receiver pairs in a 1000×1000

square on a plane. These links have length at least 0 and at most L, which is a parameter. Precision for all involved numbers is *double*. Two example networks generated with the two models are depicted in Figure 1.

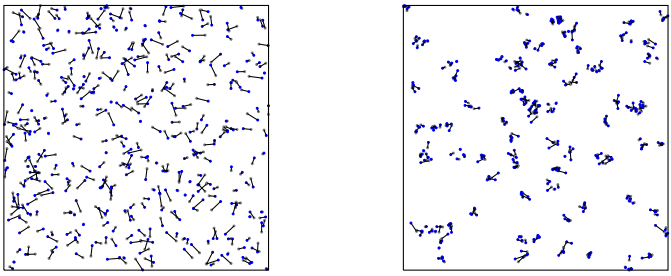


Fig. 1. Two example instance with 400 requests and $L = 50$. Left: Unclustered Network Right: Clustered Network with exponential distribution, $\gamma_c = \gamma_r = 0.2, c = n/5$.

The simplest way of construction is an *unclustered network*. Here, we first determine for each link independently the position of the sender node s uniformly in the plane. In the second step, for each sender we place the corresponding receiver r independently. This is performed by selecting a vector k by determining an angle δ uniformly from $[0, 360°)$ and a distance d uniformly from $[0, L)$. The receiver r is then placed by $r = s+k$ if this point lies inside the given 1000×1000 plane. Otherwise, this step is repeated.

Real-world networks typically show clustering effects and that most distances are comparably short. These aspects are taken into account in *clustered networks with exponential distribution* (see, e.g., [17]). We first select c cluster centers independently uniformly at random in the plane. Afterwards, in the vicinity of each cluster center n/c sender nodes are placed. For each sender the respective receiver is then placed similarly in the vicinity of its sender. In both steps, an angle and a distance are selected independently. The angle is again chosen uniformly at random from $[0, 360°)$. The distance between the cluster center and the sender is determined using an exponential distribution with mean $\gamma_c \cdot L$. For the distance between the sender and the receiver node, we use an exponential distribution with mean $\gamma_r \cdot L$. If a node lies outside the plane or one of the distances exceeds L, the step is repeated.

All of the results presented in this paper were obtained for clustered networks with $L = 50$, $\gamma_c = \gamma_r = 0.2$, and $c = n/5$. The SINR parameters were set to $\alpha = 4$ and $\beta = 1$. In terms of absolute numbers the choice of the network model and its parameters make a large difference in our simulations. However, the relative performances seem to be robust against changes in the model or in the parameters. A further discussion of this issue is deferred to the full version.

3 Comparison of Power Assignments

In order to benchmark different power assignments, we compare the theoretical optima in the respective cases. To do so, we solve the capacity-maximization problem with a fixed power assignment as integer linear program (ILP) as follows. For each link ℓ, we have an indicator variable x_ℓ being assigning the value 1 (accepted) or 0 (rejected). The objective is to maximize the sum of all indicator variables.

$$\max \sum_{\ell \in \mathcal{R}} x_\ell \tag{1a}$$

$$\text{s.t. } \frac{p(\ell)}{d(s,r)^\alpha} + M(1 - x_\ell) \geq \beta \left(\left(\sum_{\ell' = (s',r') \neq \ell} \frac{p(\ell')}{d(s',r)^\alpha} x_{\ell'} \right) + N \right) \quad \text{for all } \ell = (s,r) \in \mathcal{R} \tag{1b}$$

$$x_\ell \in \{0,1\} \quad \text{for all } \ell \in \mathcal{R} \tag{1c}$$

The SINR constraint modeled in Equation (1b) has to be satisfied by each active request ℓ. That is, depending on the binary variable x_ℓ the respective constraint has to be fulfilled or not. To effect this behavior and to receive a linear program so called *big M formulations* are used, setting M to a sufficiently large constant. To ensure numerical robustness, the input for the LP solver was expressed using *indicator constraints*. Internally, the LP solver transforms these constraints and sets suitable values for M.

To optimize over all possible power assignments, we use the following mixed integer linear program (MILP).

$$\max \sum_{\ell \in \mathcal{R}} x_\ell \tag{2a}$$

$$\text{s.t. } \frac{p_\ell}{d(s,r)^\alpha} + M \cdot (1 - x_\ell) \geq \beta \left(\left(\sum_{\ell' = (s',r') \neq \ell} \frac{p_{\ell'}}{d(s',r)^\alpha} \right) + N \right) \quad \text{for all } \ell = (s,r) \in \mathcal{R} \tag{2b}$$

$$0 \leq p_\ell \leq p_{\max} x_\ell \quad \text{for all } \ell \in \mathcal{R} \tag{2c}$$

$$x_\ell \in \{0,1\} \quad \text{for all } \ell \in \mathcal{R} \tag{2d}$$

In this program, we have for each request ℓ a variable p_ℓ specifying the respective transmit power. Again, big M formulations ensure that the SINR constraint is satisfied for each link having $x_\ell = 1$. If $x_\ell = 0$, the SINR constraint does not need to be fulfilled and the power is set to 0.

3.1 Simulation Setting

Our simulations were run with the optimization software CPLEX v12.2. In order to eliminate effects of particular networks or parameters, we repeated all simulations with differently generated networks, with and without clustered structure. Furthermore, we applied multiple parameter settings. Although the absolute

number of links that can be scheduled depends greatly on network and parameter settings, the relation between the optima does not change significantly. We ran 10 simulations for each setting and calculated the average result.

With the presented models for the non-convex optimization we computed the optimal subset of links with respect to arbitrary power assignments and also with respect to uniform, linear, and square-root power assignments. For fixed power assignments we were able to calculate optimal results regarding the maximum capacity problem up to a network size of 1600 requests.

The problem gets significantly more involved when attempting to optimize the power assignment. With an increasing number of transmissions it gets rapidly harder to reduce the remaining integer gap. Thus, we set a time limit of 3 hours or accepted the results as the optimal results when the integer gap reduced to a value of less than 3 %. We were able to generate optimal results for networks with less than 200 requests. For networks consisting of more than 200 requests the time limit took effect. That is, the obtained solution is more than 3 % worse than the fractional upper bound at this point. However, we use these solutions for our considerations. This is due to the fact that choosing a much larger time limit could not improve the qualities of the solutions significantly. Furthermore, the solutions obtained up to this point still turned out to be significant since they outperformed the maximal number of scheduled requests achieved with a fixed power assignment.

3.2 Simulation Results

Figure 2 shows the results of the comparison of uniform, linear, and square-root power assignments with the optimized one for network sizes of up to 800 requests. Network instances consisting of 1600 requests did not yield meaningful outputs

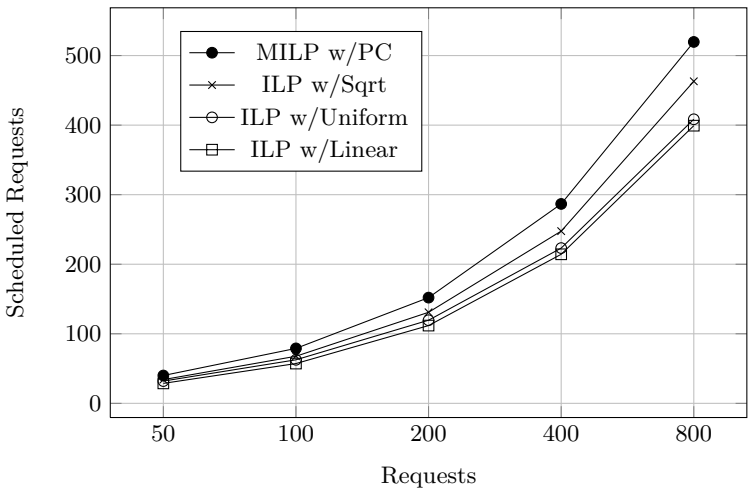


Fig. 2. Comparison of Theoretical Optima (10 test runs)

for the case of power control. The results are given for "standard" parameters but, as further explained in the full version, the ranking of the algorithms and the behavior remained consistent for all tested parameters.

As mentioned in the previous section, the displayed values for the mixed integer linear program are only the best solutions computed within 3 hours. Although these are not necessarily the theoretical optima, they already reveal the potential given by the use of power control. For example, given a clustered network with 800 requests, it can be observed that a linear or uniform power assignment can select about half of the requests on average. The number of requests selected with power control is more than 500 requests on average. Hence, using power control for this parameter setting allows a performance gain of $15 - 17\%$ compared to uniform power assignments. Square-root power assignments are partly able to exploit this potential of power control. The network capacities generated with square-root power assignments are significantly larger than the capacities achieved with a uniform or linear power assignment. Thus, the assumed theoretical advantage of square-root power assignments could clearly be confirmed.

Table 1 presents the remaining integer gap for the MILP approach on a typical network instance. A further interesting fact is presented by the last column of Table 1 giving the ratio between the best achieved ILP solution with a uniform power assignment and the best achieved integer solution of MILP. This ratio stays constant at about 80 % also for larger networks. This suggests that the obtained solutions are nearly optimal ones for the MILP approach.

4 Evaluation of Approximation Algorithms

For our simulations, we implemented the following three algorithms, each being a constant-factor approximation for the respective optimum (see Section 1.2): The algorithm by Kesselheim [12] using power control, abbreviated by *Kess*, the one by Halldórsson and Wattenhofer [11] using uniform transmission powers (HW), and the one by Halldórsson and Mitra [10] using square-root power assignments (HM).

The three employed approximation algorithms have the same underlying working principle. They examine the requests of the network in order of increasing length. A request ℓ' is (tentatively) selected if it satisfies a condition of the form $\sum_{\ell \in \mathcal{L}} w(\ell, \ell') \leq W$, where $\mathcal{L}$ is the set of previously selected links. The weight $w(\ell, \ell')$ and the constant bound W depend on the actual approximation algorithm. Having made this tentative selection, the final solution is computed by

Table 1. Average Optimal Results (10 test runs, time limit: 3h)

Requests	ILP uniform	MILP	MILP + Gap	Ratio $^{\text{ILP}}/_{\text{MILP}}$
50	32.0	40.0	40.0	0.80
100	62.6	79.0	79.8	0.79
200	119.5	152	188.3	0.79
400	223.2	286.7	388.1	0.78
800	408.4	519.6	766.9	0.79

choosing a subset of the selection or assigning transmit powers. For the purpose of proving the desired approximation factor, the value W of the selection constraint is chosen very conservatively. In random instances, slight relaxations still result in feasible solutions for most situations. Thus, we implemented an additional binary search to obtain an appropriate bound W. This adapted bound admits better results which are still feasible.

Furthermore we implemented the so-called *MinLoss* and *MaxLoss* heuristics, which are the simplest greedy algorithms for the problem of approximating the capacity-maximization problem with a fixed power assignment. However, they only yield a poor worst-case performance and no non-trivial approximation factors can be proven. The requests are considered in order of increasing (MinLoss) or decreasing (MaxLoss) path loss. This is equivalent to examining the requests in order of increasing request lengths since the path-loss exponent α is fixed. A request ℓ is added to the set $\mathcal{L}$ if all involved SINR constraints are fulfilled afterwards. That is, not only the condition for ℓ is checked but also the ones for all previously added links. For this reason, for the MinLoss or MaxLoss algorithm $O(n^2)$ times the interference between a request and the already assigned requests has to be calculated. In contrast, the remaining approximation algorithms always have to check a single constraint, resulting in $O(n)$ calculations of a constraint. This means that in terms of the required calculation time the MinLoss and MaxLoss algorithms need an additional factor of $O(n)$.

We implemented both heuristics, MinLoss and MaxLoss, both with a uniform and a square-root power assignment. In the following they are referred to as *MinSqrt*, *MinUni* and *MaxSqrt*, *MaxUni*, respectively.

We used Java for our implementations and analyzed network instances with up to 1600 requests.

Simulation Results. The results of 10 test runs are given in Figure 3, comparing the approximation results to the uniform optimum. Interestingly, the uniform optimum achieves results that are similar to MinSqrt, HM and Kess. The previous section showed that the uniform optimum is outperformed by the optima produced with a square-root power assignment or power control. Thus there is still room for improvement for these algorithms. For the sake of clarity, the uniform variants of MinLoss and MaxLoss are not displayed because they are outperformed by their square-root variants.

The weak performance of the MaxLoss heuristic for larger network instances can easily be explained. The MaxLoss heuristic initially examines the longest requests. Due to the fact that with a larger distance between sender and receiver the transmission has to be performed with a higher power value, also the arising interference increases. The reason for the good performance of the MinLoss heuristic, also with a uniform power assignment, is exactly the one why its worst-case performance is very poor. In contrast to the approximation algorithms, that are very conservative, it uses the original SINR constraints. Nevertheless, it also needs a larger computation time as already stated.

Furthermore, we can observe that the HW algorithm achieves much weaker results. Compared to Kess and HM, this is due to the fact that it uses uniform

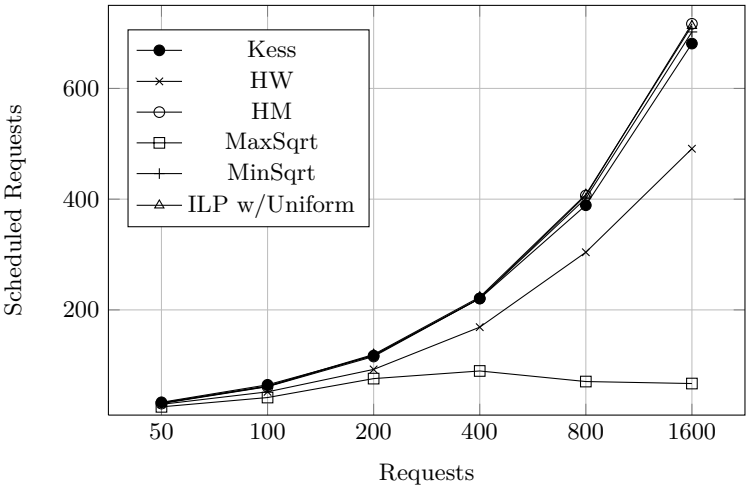


Fig. 3. Comparison of Approximations and Uniform Optimum (10 test runs)

powers, which were shown to be inferior in the previous section. So the weaker performance of a uniform or linear power assignment for the calculation of the maximal network capacity is also revealed by our simulations with the approximation algorithms. This can be observed as well when only comparing MinSqrt to MinUni or MaxSqrt to MaxUni. The variants using a square-root power assignment always outperform their uniform variants. We can even observe that the approximation algorithms using a square-root power assignment or power control, are able to outperform the uniform or linear optimum for some of the smaller instances and can keep up to it in all instances. This means that these algorithms already achieve better results than any algorithm using uniform powers, even one with unlimited computational powers.

However, theoretically, also between these algorithms there were large differences. The Kess algorithm can be shown to guarantee better results than any algorithm using square-root power assignments. Up to now, this cannot be verified by our simulations, where both the Kess and the HM algorithms perform very similarly. Thus, there is still further potential by the use of power control, which is not significantly presented by the implemented algorithms.

A last aspect to be mentioned are running times, as the study of approximation algorithms is motivated by the fact that they ensure a polynomial running time. This discrepancy can also be observed in our simulations. While the running time does not differ significantly for smaller network instances, the required computation time for a network instance consisting of 800 requests differs by a factor of more than 100: Running the approximation algorithms still takes less than a second, whereas solving the ILP takes a minute. The MILP approach using power control does not even complete within days.

Latency Minimization. For our simulations, we focused on the capacity-maximization problem because it is particularly simple to state as an ILP or as and MILP. Practical works, in contrast, often consider the latency-minimization, in which one has to calculate a schedule that serve all requests at least once. Unfortunately, existing approaches [13,6,2] are not able to solve the problem exactly in reasonable time. Furthermore, most approximation algorithms for latency minimization actually solve recursively maximize the use of the first slot.

Nevertheless, we implemented these algorithms as well and observed a similar ranking as for capacity maximization with one striking difference. The MaxLoss heuristics are able to catch up to their MinLoss correspondence. This is due to the fact that the processing order is not as important as for capacity maximization because all requests have to be served anyway.

Further details on these results can be found in the full version. However, dealing with latency minimization mostly remains an interesting topic for future research, because finding the minimum-latency solution within acceptable time still remains an open problem.

5 Conclusion

Our simulations are able to support a number of theoretical findings from worst-case analysis. In particular, square-root power assignments appear to be not only good in theory but also in practice. They originate from theoretical considerations, in which it was shown that they are superior to uniform or linear power assignments. Our simulations confirm this observation. Thus they are an easily implementable alternative that is at least partially able to profit from power control. However, one can get better performances by optimizing the power assignment – both in theory and in simulations.

Comparing the approximation algorithms and heuristics, we can again observe the effects of power control. Algorithms using different transmission powers are in general superior to the ones using uniform power assignments. In particular, we have shown that the developed approximation algorithms can compete with the other approaches in terms of the solution quality. However, they have the advantage that there are guarantees on the performance in any network. Furthermore, structurally they are as simple as the heuristics. Their running times are only quadratic in the network size while the heuristics need cubic running times. Hence, developing approximation algorithms seems to be the right direction.

A last point to be mentioned is that the purpose of the simulations was to study the given algorithms on randomly generated networks in contrast to the worst-case assumptions used in algorithmic theory. This brings about that we neglected modeling issues and carried out the simulations within the same model. It remains an open problem to verify our results in more advanced simulation environments or even in real-world experiments.

References

1. Andrews, M., Dinitz, M.: Maximizing capacity in arbitrary wireless networks in the SINR model: Complexity and game theory. In: Proc. 28th INFOCOM, pp. 1332–1340 (2009)
2. Charalambous, T., Klerides, E., Wiesemann, W.: On the transmission scheduling of wireless networks under SINR constraints. CUED/F-INFENG/TR.649 (2010)
3. Cruz, R., Santhanam, A.V.: Optimal routing, link scheduling and power control in multihop wireless networks. In: Proc. 22nd INFOCOM, pp. 702–711 (2003)
4. ElBatt, T.A., Ephremides, A.: Joint scheduling and power control for wireless ad hoc networks. IEEE Transactions on Wireless Communication 3(1), 74–85 (2004)
5. Fanghänel, A., Kesselheim, T., Räcke, H., Vöcking, B.: Oblivious interference scheduling. In: Proc. 28th PODC, pp. 220–229 (2009)
6. Fu, L., Liew, C., Huang, J.: Joint power control and link scheduling in wireless networks for throughput optimization. In: Proc. IEEE International Conference on Communications (ICC), pp. 3066–3072 (2008)
7. Goussevskaia, O., Wattenhofer, R., Halldórsson, M.M., Welzl, E.: Capacity of arbitrary wireless networks. In: Proc. 28th INFOCOM, pp. 1872–1880 (2009)
8. Gupta, P., Kumar, P.R.: The capacity of wireless networks. IEEE Transactions on Information Theory 46, 388–404 (2000)
9. Halldórsson, M.M.: Wireless Scheduling with Power Control. In: Fiat, A., Sanders, P. (eds.) ESA 2009. LNCS, vol. 5757, pp. 361–372. Springer, Heidelberg (2009)
10. Halldórsson, M.M., Mitra, P.: Wireless capacity with oblivious power in general metrics. In: Proc. 22nd SODA, pp. 1538–1548 (2011)
11. Halldórsson, M.M., Wattenhofer, R.: Computing wireless capacity (2010) (unpublished manuscript)
12. Kesselheim, T.: A constant-factor approximation for wireless capacity maximization with power control in the SINR model. In: Proc. 22nd SODA, pp. 1549–1559 (2011)
13. Kompella, S., Wieselthier, J.E., Ephremides, A.: Revisiting the optimal scheduling problem. In: Proc. 42nd Annual Conference on Information Sciences and Systems (CISS), pp. 492–497 (2008)
14. Moscibroda, T., Oswald, Y.A., Wattenhofer, R.: How optimal are wireless scheduling protocols? In: Proc. 26th INFOCOM, pp. 1433–1441 (2007)
15. Moscibroda, T., Wattenhofer, R.: The complexity of connectivity in wireless networks. In: Proc. 25th INFOCOM, pp. 1–13 (2006)
16. Wan, P.-J., Ma, C., Tang, S., Xu, B.: Maximizing Capacity with Power Control under Physical Interference Model in Simplex Mode. In: Cheng, Y., Eun, D.Y., Qin, Z., Song, M., Xing, K. (eds.) WASA 2011. LNCS, vol. 6843, pp. 84–95. Springer, Heidelberg (2011)
17. Waxman, B.M.: Routing of multipoint connections. IEEE Journal on Selected Areas in Communications 6(9), 1617–1622 (1988)
18. Zander, J.: Performance of optimum transmitter power control in cellular radio systems. IEEE Transactions on Vehicular Technology 41(1), 57–62 (1992)

Approximating Barrier Resilience for Arrangements of Non-identical Disk Sensors

David Yu Cheng Chan and David Kirkpatrick

Department of Computer Science,
University of British Columbia, Vancouver B.C., Canada

Abstract. Let $\mathcal{A}$ be an arrangement of n sensors constituting a barrier between two regions S and T. The *resilience* of $\mathcal{A}$ with respect to S and T, denoted $\rho(\mathcal{A}, S, T)$, is defined as the number of sensors in $\mathcal{A}$ that must be removed in order for there to be an $S - T$ path that is not detected by any sensor. We introduce the *Multi-Path Algorithm* (MPA) and show that it guarantees a 2-approximation of $\rho(\mathcal{A}, S, T)$ in time polynomial in n when sensors are unit disks in a two dimensional plane; this tightens to a 1.5-approximation when S and T are moderately well-separated. We also define a generalization of $\rho(\mathcal{A}, S, T)$ denoted $\rho_c(\mathcal{A}, S, T)$, which is the resilience of the barrier if regions covered by more than c distinct sensors in the original arrangement are treated as inaccessible. Then when the unit size constraint is relaxed, we prove that the MPA can still guarantee a 2-approximation of $\rho_c(\mathcal{A}, S, T)$ for any constant c in time polynomial in n for arrangements of approximately equal-sized sensors.

Keywords: barrier coverage, resilience, wireless sensor networks.

1 Introduction

Since the introduction of Wireless Sensor Networks, various notions of sensor coverage have been investigated for their potential utility, to the extent that many works in the literature provide overviews of the subject [1–3]. In this paper, we focus on barrier coverage problems in two dimensions.

In general, the input is an arrangement $\mathcal{A}$ of n sensors forming a barrier, a starting region S, and a target region T. The question is then to determine how impermeable the barrier is with respect to covering $S - T$ paths.

Many concepts of barrier impermeability have been proposed, each with their advantages and disadvantages. One idea is to maximize breach, or distance between sensors and paths, which is used by notions such as *average breach paths* [4] and *maximal breach paths* [1, 5–8]. Another idea is to minimize the exposure over all points on the path, used by *minimal exposure paths* [9–12].

In contrast to these measurements which may have large changes in value upon the failure of even a single sensor, other notions of barrier impermeability have been defined to capture the idea of robustness in the face of sensors which are not immune to failure.

In particular, the work by Kumar *et al.* [13, 14] considers the case where each sensor is associated with some detection region. If a path P intersects the

A. Bar-Noy and M.M. Halldórsson (Eds.): ALGOSENSORS 2012, LNCS 7718, pp. 42–53, 2013.

region of some sensor s_i, the start and end of the intersection is considered a *sensor entry* and *exit* respectively. The subpath associated with the intersection is considered a *sensor crossing*, and P is said to cross s_i. Note that P may have multiple s_i crossings, and in general no sensors cover S or T so the number of entries, exits and crossings are equivalent. $\mathcal{A}$ is said to provide k-barrier coverage if k is the minimum over all $S-T$ paths of the number of distinct sensors crossed. This measure decreases by at most 1 for each sensor that fails, and detection of any $S-T$ path can be guaranteed as long as fewer than k sensors fail. Kumar *et al.* [13, 14] presented a method for determining the exact strength of barriers when sensors are disks in a two-dimensional open belt region. These open belt regions are defined as strip-like regions with no holes such that every $S-T$ path must cross the region without intersecting the strip ends.

Bereg and Kirkpatrick [15] built on this definition by defining two notions of barrier impermeability: *thickness*, denoted $\tau(\mathcal{A}, S, T)$ and defined as the minimum over all $S-T$ paths of the number of sensor entries, and *resilience*, denoted $\rho(\mathcal{A}, S, T)$ and defined as the minimum number of sensors that must be removed from $\mathcal{A}$ to make $\tau(\mathcal{A}, S, T) = 0$. An $S-T$ path is *resilience-optimal* if and only if it intersects exactly $\rho(\mathcal{A}, S, T)$ distinct sensors. The key difference between the notions is that $\rho(\mathcal{A}, S, T)$ counts each sensor at most once, whereas $\tau(\mathcal{A}, S, T)$ will count a sensor multiple times if it is entered multiple times. For instance, for both arrangements illustrated in Fig. 1, $\rho(\mathcal{A}, S, T) = 2$, while $\tau(\mathcal{A}, S, T) = 4$. Effectively, a k-barrier covered region has $\rho(\mathcal{A}, S, T) = k$.

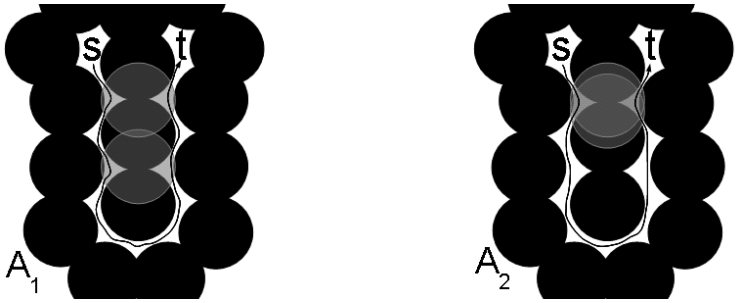


Fig. 1. Arrangement $\mathcal{A}_1$ and $\mathcal{A}_2$. Black regions indicate heavy sensor coverage, grey circles indicate sensors, while the black path from S to T indicates the resilience-optimal path for each arrangement.

Bereg and Kirkpatrick [15] considered arbitrary arrangements in the plane. They observed that any path in an arrangement $\mathcal{A}$ can be represented by a walk in the dual graph $\hat{\mathcal{A}}$ constructed by transforming the faces of $\mathcal{A}$ into vertices which are connected by edges if and only if they were adjacent faces. This is illustrated by Fig. 2. Given regions S and T, two vertices s' and t' representing the respective regions are then added to the dual graph and connected to every vertex representing a face inside S and T respectively. Thus all $S-T$ paths in $\mathcal{A}$ correspond to $s'-t'$ walks in $\hat{\mathcal{A}}$.

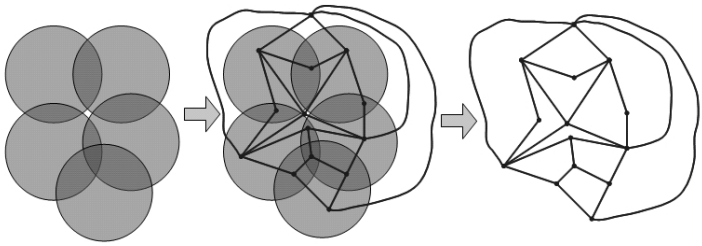


Fig. 2. Constructing the Dual Graph $\hat{\mathcal{A}}$ of an Arrangement $\mathcal{A}$

With this dual graph, the problem of determining $\tau(\mathcal{A}, S, T)$ can be reduced to a graph shortest-path problem by assigning weights $w(i, j)$ to directed edges corresponding to the number of sensors entered. Thus any of the standard graph shortest-path algorithms will suffice for thickness. In contrast, the resilience is more difficult to compute since each sensor is counted at most once regardless of how many times it is entered.

For any given path P and any given sensor s_i crossed by P, two points on the boundary of s_i are considered consecutive if there exists an arc on the boundary of s_i which connects the two points and is not intersected by P. We say a crossing is *peripheral* if its endpoints are consecutive, otherwise it is *interior*. P is considered a *proper traversal* if P is a non-self-intersecting $S-T$ path whose every crossing is peripheral. The main motivation for distinguishing proper traversals is the following:

Lemma 1 (Bereg and Kirkpatrick [15]). *For arrangements of unit disk sensors, there exists a resilience-optimal proper traversal that crosses each sensor at most* 3 *times. This is reduced to* 2 *times if* S, T *is separated by* $> 2\sqrt{3}$.

This means $\tau(\mathcal{A}, S, T)$ forms a 3 and 2-approximation of $\rho(\mathcal{A}, S, T)$ in these cases respectively. Bereg and Kirkpatrick [15] also noted that if a graph shortest-path algorithm could be modified to detect when sensor crossings are associated with previously crossed sensors, it could discount the weight of the path appropriately to derive $\rho(\mathcal{A}, S, T)$.

Furthermore, because all crossings on a proper traversal are peripheral, each crossing can be assigned a relative *orientation*, *positive* or *negative* if the non-intersected arc from the entry point to the exit point on the boundary of the sensor goes clockwise or counter-clockwise respectively. For instance, all crossings in Fig. 1 are negative. Using this orientation notation, Bereg and Kirkpatrick [15] observed that certain ordered sequences of crossings with specified orientations are forbidden in the sense that they cannot appear on proper traversals. This allowed them to develop an algorithm with limited discount detection ability and argue that it detects $\frac{1}{3}$ of the discounts in the well-separated S, T case, effectively guaranteeing a $\frac{5}{3}$-approximation of $\rho(\mathcal{A}, S, T)$.

In this paper, we extend previous work in three significant ways. The first is to augment a memoryless shortest-path algorithm with greater discount

detection ability while maintaining time complexity polynomial in n, resulting in the development of the *Multi-Path Algorithm* (MPA) discussed in Sect. 3.

The second extension is to relax the unit size constraint enforced on disk sensors. While for some arrangements of non-identical disk sensors all resilience-optimal $S - T$ paths are not proper traversals, we note that there are many settings where the existence of a resilience-optimal proper traversal can still be guaranteed, such as when disk sizes are close to uniform. Thus when the unit size constraint is relaxed, the disk sensors are considered approximately equal-sized if and only if there exists a resilience-optimal proper traversal. The key difference that arises from relaxing the constraint is that the crossing limits in Lemma 1 do not hold. In fact, it is possible to construct arrangements which force all resilience-optimal proper traversals to cross a sensor an indefinite number of times if the disk sizes differ even slightly.

Intuitively, this is a result of the fact that crossings of the same sensor must be isolated from one another by other sensors that the path avoids, otherwise there exists an alternative path that just goes directly between two non-isolated crossings, and this alternative path would cross at most the same number of distinct sensors as the original while making fewer crossings. Smaller sensors can be used to isolate an indefinite number of peripheral crossings from one another, thus allowing arrangements to force an indefinite number of recrossings, but isolating an interior crossing is more difficult since there must be avoided sensors on both sides of the crossing. Note that we still enforce the constraint that sensors must be disk-shaped, as work by Tseng and Kirkpatrick [16, 17] indicates that resilience approximation may be NP-hard in general for sensors with non-symmetric shape.

The third extension is based on the observation that barrier resilience, as it has been conventionally defined, provides a measure of *cumulative* sensor coverage for points on $S - T$ paths. It is also interesting to consider *instantaneous* sensor coverage, as well as combinations/tradeoffs of these notions. To this end, we can say a sensor arrangement forms a (c, k)-barrier with respect to regions S and T if every $S - T$ path P either (i) has a $(c + 1)$-*detected* face (a face covered by $c + 1$ or more sensor regions), or (ii) intersects at least k sensor regions over its entire length. The arrangements illustrated in Fig. 1 are both 2-barriers and $(2, 2)$-barriers, but $\mathcal{A}_2$ is a $(1, \infty)$-barrier since every $S - T$ path must have a 2-detected face, whereas $\mathcal{A}_1$ is only a $(1, 2)$-barrier.

This leads to a natural generalization of resilience that we call *density-c* (or D_c) resilience, denoted $\rho_c(\mathcal{A}, S, T)$, which considers only $(c + 1)$-*evasive* $S - T$ paths (paths that avoid all $(c + 1)$-detected faces). Note that computing D_0 resilience amounts to determining if S and T lie in the same face of the arrangement, whereas computing D_c resilience for $c \geq n$ is equivalent to computing the standard resilience. Thus it suffices to consider only cases where $1 \leq c \leq n$.

In Sect. 2, we describe representations of proper traversals that allow us to capture their key features with respect to discount detection. Then in Sect. 3, we discuss the MPA and prove that its improved discount detection abilities allows us to tighten the previously best approximation ratios of the unit disk

scenarios from 3 and $\frac{5}{3}$ to 2 and 1.5 respectively. Finally in Sect. 4, we prove that the MPA guarantees a poly-time 2-approximation of $\rho_c(\mathcal{A}, S, T)$ for any constant c for approximately-equal sized disk sensor arrangements. Due to space constraints, many details and proofs have been omitted. Readers are invited to read the associated thesis [18] for these and other related results.

2 Optimal Path Representations

Any path which is a proper traversal can be represented as a timeline by imagining some entity following the path from S to T. We represent this timeline by drawing a horizontal line segment on a two-dimensional plane and drawing intervals on the timeline to represent the sensor crossings for each sensor. Note that since paths may cross multiple sensors simultaneously, the crossings intervals may overlap and any one interval may be composed of several subintervals corresponding to faces in the arrangement.

Each discount associated with the path comes from recognizing that some pair of crossings on the path is associated with the same sensor s_i. One way to certify this is to examine a face in each crossing and observe that they are both covered by s_i. We can represent this certificate by drawing a smooth arc, referred to as a bracket, with endpoints connected to a face in each crossing. As all discounts associated with a sensor are detected when all crossings of the sensor are recognized as sharing the same sensor, the discounts are fully accounted for when the brackets form a spanning tree connecting every crossing of the sensor.

A set of brackets is considered a *full bracketing* if there is exactly one spanning tree for each sensor with no extra brackets. This is full in the sense that detecting and combining all the associated discounts when determining the weight of the path will derive the exact number of distinct sensors crossed. Note that this means timelines may have more than one full bracketing, but all full bracketings associated with the same path must have the same number of brackets. Any subset of a full bracketing is considered a *partial bracketing*.

A bracketing will be called *planar* if its brackets can be drawn such that no two brackets intersect each other and no bracket intersects the timeline. It turns out that the crossing restrictions of proper traversals imply the existence of planar full bracketings because of the forbidden sequences of crossings with specified orientations discussed in [15, 18]. [18] also provides a proof of the following:

Lemma 2. *Every proper traversal has at least one planar full bracketing. Furthermore, if the proper traversal has no overlapping crossing intervals, every full bracketing is planar.*

3 Multi-Path Algorithm

The MPA is a dynamic programming algorithm that operates in a sequence of phases. It takes an integer parameter $X > 0$, and in each phase $y \geq 0$, a $2X$-dimensional array of size $|V|$ in every dimension is created. Each array entry

$d_y[i_1, j_1; i_2, j_2; ...; i_X, j_X]$ represents a sequence of X paths referred to as an X-Path, where the x-th path in the sequence is an $i_x - j_x$ path for all $1 \leq x \leq X$. The entry stores only a single weight, corresponding to the smallest upper bound that the MPA has found so far on the minimum number of distinct sensors crossed by the set of paths, with discounts made on the assumption that these paths will eventually be merged into a single path which includes them as subpaths in sequential order. For convenience, the MPA with parameter X is referred to as the X-Path Algorithm.

In phase 0, each entry representing a set of X length-0 paths from a vertex to itself is assigned 0, each entry representing an edge in $\hat{\mathcal{A}}$ and a set of $X - 1$ length-0 paths at the end of the edge is assigned the weight of the edge, and all other entries are assigned ∞.

Let $\alpha \oplus \beta = \gamma$ if the X-Paths represented by $d_{y-1}[\alpha]$ and $d_{y-1}[\beta]$ share endpoints such that they can be merged into a single X-Path that has the endpoints specified by $d_y[\gamma]$. Each phase $y > 0$ assigns $d_y[\gamma] = \min_{\alpha \oplus \beta = \gamma}(d_{y-1}[\alpha] + d_{y-1}[\beta] - \text{Discounts}(\alpha, \beta))$ for all γ, where $\text{Discounts}(\alpha, \beta)$ denotes the number of discounts we can make when merging the $2X$ paths in $d_{y-1}[\alpha]$ and $d_{y-1}[\beta]$ into X paths that match $d_y[\gamma]$. These discounts are made only by considering the endpoints of the child entries.

In addition, because it is always possible to merge with an X-Path composed of only length-0 paths, array entry values can never increase. A new phase is started if any entry is smaller in d_y than in d_{y-1}, and the algorithm only terminates when all values in the array stabilize. The array entry $d_Y[s', t'; t', t'; ...; t', t']$ where the first path in the X-Path is an $s' - t'$ path and the rest are 0-length paths at t' is then read for the smallest upper bound on $\rho(\mathcal{A}, S, T)$. Observe that to consider only $(c + 1)$-evasive paths for $\rho_c(\mathcal{A}, S, T)$, it suffices to remove all vertices in $\hat{\mathcal{A}}$ which represent $(c+1)$-detected faces. Full details on the MPA are presented in [18].

Theorem 1. *If $\mathcal{A}$ is an arrangement of n disk sensors of arbitrary sizes, the X-Path Algorithm terminates in $O(n^{10X+1})$ time.*

Proof. $|V|$ is $O(n^2)$ for disk sensors and the number of distinct values an entry can have is $n+2$. Non-terminal phases must reduce the value of at least one entry and there are $|V|^{2X}$ entries in each array, so the number of phases is $O(n^{4X+1})$. The number of possible merge steps per phase is $O(n^{6X})$, so the total time complexity is $O(n^{10X+1})$. □

Let P be any $i - j$ path and $\mathcal{D}$ be a binary tree that uses the array entries of the MPA as its nodes. We say $\mathcal{D}$ is a *valid derivation tree* for P if and only if the following constraints hold:

1. The root of $\mathcal{D}$ is the array entry in d_Y which stores the weight for $i - j$ paths. In other words, the array entry must be $d_Y[i, j; j, j; ...; j, j]$, where the first path in the X-Path is P and the rest are 0-length paths at j.
2. For each node $d_y[\gamma]$ in $\mathcal{D}$:
   (a) The child nodes of $d_y[\gamma]$ must be in d_{y-1}.

(b) If $d_y[\gamma]$ has two child nodes $d_{y-1}[\alpha]$ and $d_{y-1}[\beta]$, then $\alpha \oplus \beta = \gamma$.
(c) If $d_y[\gamma]$ has one child node, then the child node must be $d_{y-1}[\gamma]$.
(d) If $d_y[\gamma]$ has no child nodes, then $y = 0$.

3. Performing the merge steps associated with every node in $\mathcal{D}$ from the bottom to the top can produce the path P.

Theorem 2. *Given any valid derivation tree $\mathcal{D}$ for a path P, the array entry associated with any node of $\mathcal{D}$ must be assigned some value by the X-Path Algorithm which is no more than the value assigned to it by performing the merge steps associated with $\mathcal{D}$ starting from array d_0.*

Proof. The algorithm attempts every possible merge step in each phase and stores the minimum, so it is not possible for it to perform worse than any valid derivation tree in any array.

While the exact details of the discounting scheme employed by the MPA are complicated, the main gist of it is that the endpoints of X-Paths are checked for discounts during each merge step. In Sect. 2, we noted that brackets on a proper traversal P are effectively instructions on methods for detecting discounts, in that checking the endpoints of brackets allows one to recognize that two crossings share the same sensor. Thus, if an array entry has path endpoints that match some bracket, the algorithm will be able to detect the discount associated with the bracket. However, note that this array entry must actually be used in constructing P in order for its discount to be factored into the weight calculated for P. Thus a bracketing on P is considered fully detectable only if the discount for every bracket can be simultaneously factored into the weight calculated for P.

In a valid derivation tree $\mathcal{D}$, each node is an array entry that is used to construct the path P. Thus, we say $\mathcal{D}$ *recognizes* some bracket b_i associated with P if and only if there is some node in the tree whose X-Path contains some path endpoints which match the endpoints of b_i. Effectively, every bracket recognized by $\mathcal{D}$ must have its discount detected by following the merge steps of $\mathcal{D}$. Then for any bracketing $\mathcal{B}$, we say $\mathcal{D}$ recognizes $\mathcal{B}$ if and only if it recognizes every bracket in $\mathcal{B}$. Note that this means if $\mathcal{D}$ recognizes $\mathcal{B}$, performing the merge steps of $\mathcal{D}$ will fully detect $\mathcal{B}$.

The challenge in analyzing the effectiveness of the MPA is to identify structural properties of a bracketing $\mathcal{B}$ of a path P that can be exploited in the design of a valid derivation tree $\mathcal{D}$ for P that fully detects $\mathcal{B}$. Theorem 3 and Theorem 4 below provide two such characterizations.

Theorem 3. *For any planar full bracketing $\mathcal{B}_{\text{full}}$, there exists a partial bracketing $\mathcal{B}_{\text{part}}$ with at least half the brackets in $\mathcal{B}_{\text{full}}$ such that $\mathcal{B}_{\text{part}}$ is fully detectable by a 2-Path Algorithm.*

Proof. The planarity of $\mathcal{B}_{\text{full}}$ allows it to be split into two fully detectable partial bracketings, one of which must have at least half the brackets.

Corollary 1. *If sensors are unit disks, the 2-Path Algorithm guarantees a 2-approximation of $\rho(\mathcal{A}, S, T)$ in time polynomial in n. This tightens to a 1.5-approximation if S, T is separated by $> 2\sqrt{3}$.*

Proof. Follows directly from the crossing limits of Lemma 1 by applying Theorem 3 on the planar full bracketing guaranteed to exist by Lemma 2. □

Let the *endpoint interval* of a bracket be defined as the interval on the timeline between the endpoints of the bracket. Then given a bracketing $\mathcal{B}$, the *bracket-span* of a sensor is defined as the union of endpoint intervals for all brackets associated with the sensor. We define the *ply* of $\mathcal{B}$ as the maximum over points on the timeline of the number of distinct sensors whose bracket-spans cover the point. In other words, $\mathcal{B}$ has ply k if every point on the timeline is within the bracket-spans of at most k distinct sensors.

Theorem 4. *If $\mathcal{B}$ has ply k, $\mathcal{B}$ is fully detectable by a k-Path algorithm.*

Proof. If $\mathcal{B}$ has ply k, in some sense there are locally only k distinct sensors with bracket discounts to detect at any point on the timeline. Thus we can build a valid derivation tree by sweeping from s' to t' with a k-Path, using each path to remember one sensor to ensure all discounts get detected and factored into the final weight.

4 Approximately Equal-Sized Disk Sensors

4.1 D_1 Resilience

Consider the case where we want to approximate $\rho_1(\mathcal{A}, S, T)$. We begin by noting that since $\rho_1(\mathcal{A}, S, T)$ considers only 2-evasive paths by definition, there can be no overlapping crossings.

Given any set of non-overlapping crossings from a resilience-optimal path, we begin by constructing a partial bracketing $\mathcal{B}_{\text{part}}$ where each sensor has all their crossings connected sequentially only to crossings of the same orientation. By Lemma 2, $\mathcal{B}_{\text{part}}$ is planar. This allows us to define any bracket b_i as nesting b_j if and only if all their crossings are the same orientation and the two endpoints of b_j are between the two endpoints of b_i. We then construct $\mathcal{B}_{\text{ply}}$ by removing from $\mathcal{B}_{\text{part}}$ all brackets that nest brackets.

Lemma 3. *$\mathcal{B}_{\text{ply}}$ has ply at most 2 and the difference in the number of brackets between $\mathcal{B}_{\text{ply}}$ and a full bracketing is at most the number of sensors associated with $\mathcal{B}_{\text{part}}$.*

Proof. If $\mathcal{B}_{\text{ply}}$ has ply > 2, there must be some point which is within the bracket-spans of more than 2 sensors. However, by definition that means there are more than 2 brackets covering the point, and at least 2 of them must have crossings of the same orientation. Then since the bracketing is planar, they must nest one another, which is a contradiction since we removed nesting.

$\mathcal{B}_{\text{part}}$ was constructed by sequentially connecting crossings. Since $\mathcal{B}_{\text{part}}$ is planar, the sequential brackets of the same orientation must share the same parent. This can be exploited to achieve an amortized cost of 1 bracket removed per sensor from a full bracketing.

Theorem 5. *In time polynomial in n, the 2-Path Algorithm guarantees a 2-approximation of the minimum over 2-evasive proper traversals of the number of distinct sensors crossed by the proper traversal.*

Proof. For the traversal with the minimum number of distinct sensors crossed, we can construct a bracketing $\mathcal{B}_{\text{ply}}$ which has ply 2 by Lemma 3. Then by Theorem 4, the 2-Path Algorithm fully detects all the discounts associated with $\mathcal{B}_{\text{ply}}$.

Let ρ be the number of sensors crossed by this proper traversal. Then the number of sensors associated with $\mathcal{B}_{\text{part}}$ must be at most ρ. Then by Lemma 3, we know that we miss at most ρ discounts, so the final weight of the proper traversal reported is at most 2ρ. Thus this guarantees a 2-approximation. Finally, by Theorem 1, this has time complexity polynomial in n. □

4.2 D_c Resilience for Constant c

In this section, we extend our proofs to consider $\rho_c(\mathcal{A}, S, T)$ for any constant $c > 1$. We begin by proving the following:

Theorem 6. *If the sensors crossed by a D_c-resilience-optimal proper traversal can be split into k groups such that within each group no crossings overlap, the $2k$-Path Algorithm guarantees a 2-approximation of $\rho_c(\mathcal{A}, S, T)$.*

Proof. If no crossings overlap in each group, we can use the procedure described in Sect. 4.1 to construct a partial bracketing with ply 2 for each group by Lemma 3.

Since no sensor appears in more than one group, these k partial bracketings can then be merged into a single partial bracketing, which must then have ply $2k$. Then by Theorem 4, the $2k$-Path Algorithm detects all the discounts associated with this partial bracketing. Finally by Lemma 3, this misses at most 1 bracket per sensor involved, so at most $\rho_c(\mathcal{A}, S, T)$. Thus it guarantees a 2-approximation. □

This means it suffices to show that for any constant c, there exists some constant k independent of n such that the sensors crossed by a D_c-resilience-optimal proper traversal can always be coloured by k colours so that no two sensors of the same colour have overlapping crossings. Then by Theorem 1 and Theorem 6, the MPA will guarantee a 2-approximation in time polynomial in n.

First, consider what it means for two sensors to have overlapping crossings on a proper traversal. It means there is some point on the timeline that is covered by crossings of both sensors, which means the path goes through some face in the arrangement that is covered by both sensors. Since $\rho_c(\mathcal{A}, S, T)$ only considers

$(c+1)$-evasive paths, this means two sensors can have overlapping crossings only if there exists a non-$(c+1)$-detected face in the arrangement that is covered by the two sensors.

With this in mind, we construct an undirected graph $G_c(\mathcal{A}) = (V_c(\mathcal{A}), E_c(\mathcal{A}))$ from $\mathcal{A}$ by transforming each sensor into a vertex in $V_c(\mathcal{A})$ and connecting vertices by edges if and only if there exists some non-$(c+1)$-detected face in the arrangement that is covered by the two sensors.

Corollary 2. *If there exists a vertex k-colouring on $G_c(\mathcal{A})$, the $2k$-Path Algorithm guarantees a 2-approximation of the minimum over $(c+1)$-evasive proper traversals of the number of distinct sensors crossed by the proper traversal.*

If the number of edges in $G_c(\mathcal{A})$ is at most $c'n$, the existence of a $(2c'+1)$-colouring can be proved by exploiting low degree vertices. Thus it suffices to find an upper bound on the number of edges which is linear in terms of n.

Let each sensor have a site in the arrangement positioned at the center of the sensor. For any face f, let $\check{\mathcal{S}}_f$ denote the set of sites associated with sensors that cover f. Then for arrangements of unit disk sensors, each face f is effectively a set of points which has smaller Euclidean distance to $\check{\mathcal{S}}_f$ than to any other site. For arrangements of non-identical disk sensors, this is still true if weights are added to the distances appropriately to reflect the difference in sensor radii. In other words, each non-$(c+1)$-detected face f can only exist if there is some point which has smaller weighted distance to $\check{\mathcal{S}}_f$ than to any other site.

At this point, we note the similarity of this problem to order-$\tilde{c}$ Voronoi diagrams with additive weights. These are generalizations of the standard order-1 Voronoi diagram such that each cell represents the set of points which share the same $\tilde{c}$ closest sites and the sites are weighted such that distances to each site are calculated by summing the standard Euclidean distance with the weight of the site. Thus if an order-c Voronoi diagram with the appropriately weighted distances is constructed for the sites associated with the sensors in $\mathcal{A}$, a point which has smaller weighted distance to some set of c or fewer sites than to any other site can only exist if there is a cell in the Voronoi diagram that has a superset of those sites as its closest sites.

This means that two vertices in $G_c(\mathcal{A})$ can only be connected by an edge if there exists a cell which has the two associated sites in its set of closest sites. Furthermore, each cell lists only c sites, and thus can only be used to construct $O(c^2)$ edges. In previous work, Rosenberger [19] proved that in any order-$\tilde{c}$ Voronoi diagram of $\tilde{n}$ sites with additive weights, the number of vertices, edges, and cells are all $O(\tilde{c}\tilde{n})$. Thus the number of edges in $G_c(\mathcal{A})$ must be $O(c^3 n)$.

In fact, a careful analysis allows us to reduce this to $O(cn)$. The proof of this is presented in [18]. Consequently, there must be an $O(c)$-colouring.

Corollary 3. *For any constant $c > 1$, a $O(c)$-Path Algorithm guarantees a 2-approximation of the minimum over $(c+1)$-evasive proper traversals of the number of distinct sensors crossed by the proper traversal.*

Corollary 4. *For any constant c, in time polynomial in n, the MPA guarantees a 2-approximation of the minimum over $(c+1)$-evasive proper traversals of the number of distinct sensors crossed by the proper traversal.*

5 Conclusion

For unit disk sensors, the previous best approximation ratios of the resilience were improved from a 3 and $\frac{5}{3}$-approximation to a 2 and 1.5-approximation in the general and well-separated S, T cases respectively [Corollary 1].

For non-identical disk sensors, it remains an open problem whether any algorithm could guarantee any constant-approximation of the resilience even for the slightest difference in size, even if resilience-optimal proper traversals are still guaranteed to exist. However, we proved that for any constant c, a 2-approximation is possible for the D_c resilience [Corollary 4] if D_c-resilience-optimal proper traversals exist, where the D_c resilience is a generalization of the resilience which considers only $(c+1)$-evasive paths. This is a significant contribution as the D_c resilience may be close to the standard D_n resilience in many situations even for small c as one might imagine resilience-optimal paths tend to avoid heavily covered regions and many natural settings result in low density barriers. For instance, if the density of the arrangement is low such that the number of sensors that cover each face is no more than some constant c^*, the D_n resilience becomes equivalent to the D_{c^*} resilience and can thus be 2-approximated in time polynomial in n.

However, it remains an open problem whether constant approximations of the D_n resilience are possible in time polynomial in n in the absence of such density constraints. Another open problem is whether constant approximations are possible when the constraint on disk sizes is further relaxed such that the existence of resilience-optimal proper traversals is no longer guaranteed, although some positive results have been proved in [18].

References

1. Meguerdichian, S., Koushanfar, F., Potkonjak, M., Srivastava, M.: Coverage problems in wireless ad-hoc sensor networks. In: Proceedings of the Twentieth Annual Joint Conference of the IEEE Computer and Communications Societies, INFOCOM 2001, vol. 3, pp. 1380–1387. IEEE (2001)
2. Cardei, M., Wu, J.: Coverage in wireless sensor networks. In: Ilyas, M., Mahgoub, I. (eds.) Handbook of Sensor Networks: Compact Wireless and Wired Sensing Systems, pp. 432–446. CRC Press (2005)
3. Kumar, S.: Foundations of coverage in wireless sensor networks. PhD thesis, The Ohio State University (2006)
4. DuttaGupta, A., Bishnu, A., Sengupta, I.: Maximal Breach in Wireless Sensor Networks: Geometric Characterization and Algorithms. In: Kutyłowski, M., Cichoń, J., Kubiak, P. (eds.) ALGOSENSORS 2007. LNCS, vol. 4837, pp. 126–137. Springer, Heidelberg (2008)

5. Megerian, S., Koushanfar, F., Potkonjak, M., Srivastava, M.: Worst and best-case coverage in sensor networks. IEEE Transactions on Mobile Computing 4(1), 84–92 (2005)
6. Cao, W., He, T.: Barrier coverage of wireless sensor networks based on clifford algebra. In: International Symposium on Computer Science and Computational Technology, ISCSCT 2008, vol. 2, pp. 49–52 (December 2008)
7. DuttaGupta, A., Bishnu, A., Sengupta, I.: Optimisation Problems Based on the Maximal Breach Path Measure for Wireless Sensor Network Coverage. In: Madria, S.K., Claypool, K.T., Kannan, R., Uppuluri, P., Gore, M.M. (eds.) ICDCIT 2006. LNCS, vol. 4317, pp. 27–40. Springer, Heidelberg (2006)
8. Mehta, D., Lopez, M., Lin, L.: Optimal coverage paths in ad-hoc sensor networks. In: IEEE International Conference on Communications, ICC 2003, vol. 1, pp. 507–511 (May 2003)
9. Megerian, S., Koushanfar, F., Qu, G., Veltri, G., Potkonjak, M.: Exposure in wireless sensor networks: Theory and practical solutions. Wireless Networks 8, 443–454 (2002), doi:10.1023/A:1016586011473
10. Meguerdichian, S., Koushanfar, F., Qu, G., Potkonjak, M.: Exposure in wireless ad-hoc sensor networks. In: Proceedings of the 7th Annual International Conference on Mobile Computing and Networking, MobiCom 2001, pp. 139–150. ACM, New York (2001)
11. Veltri, G., Huang, Q., Qu, G., Potkonjak, M.: Minimal and maximal exposure path algorithms for wireless embedded sensor networks. In: Proceedings of the 1st International Conference on Embedded Networked Sensor Systems, SenSys 2003, pp. 40–50. ACM, New York (2003)
12. Liu, L., Zhang, X., Ma, H.: Minimal exposure path algorithms for directional sensor networks. In: Global Telecommunications Conference, GLOBECOM 2009, November 30-December 4, pp. 1–6. IEEE (2009)
13. Kumar, S., Lai, T.H., Arora, A.: Barrier coverage with wireless sensors. In: Proceedings of the 11th Annual International Conference on Mobile Computing and Networking, MobiCom 2005, pp. 284–298. ACM, New York (2005)
14. Kumar, S., Lai, T.H., Arora, A.: Barrier coverage with wireless sensors. Wireless Networks 13, 817–834 (2007), doi:10.1007/s11276-006-9856-0
15. Bereg, S., Kirkpatrick, D.: Approximating Barrier Resilience in Wireless Sensor Networks. In: Dolev, S. (ed.) ALGOSENSORS 2009. LNCS, vol. 5804, pp. 29–40. Springer, Heidelberg (2009)
16. Tseng, K.-C.R., Kirkpatrick, D.: On Barrier Resilience of Sensor Networks. In: Erlebach, T., Nikoletseas, S., Orponen, P. (eds.) ALGOSENSORS 2011. LNCS, vol. 7111, pp. 130–144. Springer, Heidelberg (2012)
17. Tseng, K.C.R.: Resilience of wireless sensor networks. Master's thesis, The University of British Columbia (2011)
18. Chan, D.Y.C.: Approximating barrier resilience and related notions for disk sensors in a two-dimensional plane. Master's thesis, The University of British Columbia (2012)
19. Rosenberger, H.: Order-k voronoi diagrams of sites with additive weights in the plane. Algorithmica 6, 490–521 (1991)

A Unified View to Greedy Geometric Routing Algorithms in Ad Hoc Networks

Jinhee Chun, Akiyoshi Shioura, Truong Minh Tien, and Takeshi Tokuyama

GSIS, Tohoku University, Sendai 980-8579, Japan
{jinhee,shioura,tokuyama,tien}@dais.is.tohoku.ac.jp

Abstract. The main aim of this paper is to give a unified view to greedy geometric routing algorithms in ad hoc networks. For this, we firstly present a general form of greedy routing algorithm using a class of objective functions which are invariant under congruent transformations of a point set. We show that some known greedy routing algorithms such as Greedy Routing, Compass Routing, and Midpoint Routing can be regarded as special cases of the generalized greedy routing algorithm. In addition, inspired by the unified view of greedy routing, we propose three new greedy routing algorithms. We then derive a sufficient condition for our generalized greedy routing algorithm to guarantee packet delivery on every Delaunay graph. This condition makes it easier to check whether a given routing algorithm guarantees packet delivery, and the class of objective functions with this condition is closed under convex combination. We show that Greedy Routing, Midpoint Routing, and the three new greedy routing algorithms proposed in this paper satisfy the sufficient condition, i.e., they guarantee packet delivery on Delaunay graphs, and then compare the merits and demerits of these methods.

Keywords: geometric routing, ad hoc network, Delaunay graph, greedy routing.

1 Introduction

An ad hoc network is an autonomous system that does not require a pre-established infrastructure. Nodes in an ad hoc network are connected by wireless links, and the communications between nodes are often achieved by multi-hop links. With increased interests in mobile communications and the promise of convenient infrastructure-free communications, the development of large-scale ad hoc network has drawn a lot of attention and has been a subject of extensive research.

Geometric routing (also called geographic routing or position-based routing) in an ad hoc network is a technique to send a packet (or a message) from a source node to a destination node, which is done by repeatedly forwarding a packet to an appropriately chosen neighbor node. Geometric routing finds a route by using the location of the destination node and local location information, i.e., the locations of a current node and its neighbors, and does not require the

A. Bar-Noy and M.M. Halldórsson (Eds.): ALGOSENSORS 2012, LNCS 7718, pp. 54–65, 2013.

knowledge of the entire network. See, e.g., [6,8] for detailed survey of geometric routing algorithms.

The first approaches for geometric routing algorithms were developed in the 1980s, and they are based on greedy strategies (see, e.g., [3,9]); that is, they repeatedly forward a packet to a neighbor which is the "closest" to the destination node among all neighbors with respect to various criteria of "closeness." Since then, various greedy routing algorithms have been proposed in the literature.

One of the most popular and natural greedy routing algorithm is Greedy Routing by Finn [3], which forwards a packet to a neighbor with the minimum Euclidean distance to the destination node. Another popular routing algorithm is Compass Routing by Krankakis, Singh, and Urrutia [4]. The idea of Compass Routing is to forward a packet to a neighbor with the minimum angle of $\angle wvt$, where v is a current node, w is a neighbor node, and t is the destination node. A greedy routing algorithm of a different flavor is Midpoint Routing proposed by Si and Zomaya [7]. The idea of Midpoint Routing is to find a neighbor of a current node which minimizes the Euclidean distance to the midpoint of the current node and the destination. These three greedy algorithms are considered in the following discussion.

The main aim of this paper is to give a unified view to existing greedy routing algorithms. For this, we firstly present a general form of greedy routing algorithm by using a general objective function $f(w, v, t)$ defined on the set of triplets of distinct nodes; the value $f(w, v, t)$ depends only on three nodes v, w, and t representing a current node, its neighbor node, and the destination node, respectively. The generalized greedy routing algorithm repeatedly forwards a packet from a current node v to a neighbor node w minimizing the function value of $f(w, v, t)$. In particular, we introduce a class of objective functions, called *congruence-invariant objective functions*, which satisfy the condition $f(w, v, t) = f(w', v', t')$ whenever the two triangles $\triangle wvt$ and $\triangle w'v't'$ are *congruent*, i.e., $\triangle w'v't'$ can be obtained from $\triangle wvt$ by a combination of translations, rotations, and reflections. We show that various existing routing algorithms such as Greedy Routing, Compass Routing, and Midpoint Routing can be regarded as special cases of the generalized greedy routing algorithm with congruence-invariant objective functions. Moreover, inspired by the unified view of greedy routing, we propose three new greedy routing algorithms with congruence-invariant objective functions.

One of the most important factors of routing algorithms is *guaranteed delivery* of packets, and we are interested in which situation (and by which routing algorithm) packet delivery from a given source node to a given destination node is guaranteed. In general, greedy routing algorithms often fail to deliver a packet to the destination node due to the existence of a *local minimum*; local minimum is a node which has no neighbor closer to the destination node. On the other hand, it is shown that some greedy routing algorithms succeed in delivery of packets if the topology of a given ad hoc network has a nice structure. We can represent the topology of an ad hoc network by using an undirected graph $G = (P, E)$ defined on the node set P, where E is the set of all pairs of two

distinct nodes in P which can communicate to each other. It is shown in [1,4,7] that Greedy Routing, Compass Routing, and Midpoint Routing always guarantee packet delivery if the graph G corresponds to a Delaunay graph of the node set P (see Section 2 for the definition of a Delaunay graph), although the proofs of the statements are given independently in an ad hoc manner.

Aiming at providing a unified view to greedy routing algorithms with guaranteed delivery, we derive a sufficient condition for our generalized greedy routing algorithm to guarantee packet delivery on Delaunay graphs. Delaunay graph is a popular network in ad hoc network design since it has a nice property as Euclidean spanner [2] which assures a small ratio of the shortest-path length on them to the Euclidean distance, and it can be constructed in a distributed fashion.

The sufficient condition for the guaranteed delivery makes it easier to check whether a given routing algorithm guarantees packet delivery. Indeed, we show that Greedy Routing, Midpoint Routing, and the three new greedy routing algorithms proposed in this paper satisfy the sufficient condition. This provides alternative proofs for Greedy Routing and Midpoint Routing, while this implies that the new routing algorithms guarantee packet delivery on Delaunay graphs. Moreover, it is shown that these routing algorithms work on any supergraphs of the Delaunay graph of a given node set. Hence, our proposed algorithms are simple and have nice properties that are not assured in the previous ones. We also give an algebraic structure on a set of algorithms which makes it possible to obtain hybrid algorithms systematically.

The organization of this paper is as follows. Section 2 is devoted to preliminaries on fundamental concepts used in this paper. In Section 3, we propose a generalized greedy routing algorithm. We show guaranteed delivery of the algorithm on Delaunay graphs in Section 4. We compare merits and demerits of the routing algorithms in Section 5.

2 Preliminaries

Let P be a finite set of nodes. Our problem is to find a route from a given source node s to a given destination node t in an ad hoc network on P. In the following discussion, we often represent the topology of the network by an undirected graph $G = (P, E)$ on P, where E is the set of all pairs of two distinct nodes in P which can communicate to each other. We say that a node w is a *neighbor* of another node v if w and v are connected by an edge.

For simplicity we assume, throughout this paper, that nodes in P are in general position. This implies, in particular, that there are no four nodes which lie on the same circle. This assumption can be removed by using symbolic perturbation.

Voronoi diagram of P is a partition of the Euclidean plane $\mathbb{R}^2$ into polyhedral cells corresponding to nodes in P such that all nodes in a cell corresponding to $v \in P$ are closer to v than other nodes in P. *Delaunay triangulation* of P is a triangulation of P such that two nodes $u, v \in P$ are connected by a straight

line if and only if two cells corresponding to u and v have a common edge in the Voronoi diagram of P. We regard a Delaunay triangulation as an undirected graph $G = (P, E)$ on P, and call G a *Delaunay graph* of P.

The following property of Delaunay graphs is well known. For every nodes $x, y \in \mathbb{R}^2$, we denote by $d(x, y)$ the Euclidean distance between x and y, i.e., $d(x, y) = \|x - y\|_2$. A *closed disk* $D \subseteq \mathbb{R}^2$ is a set of points in $\mathbb{R}^2$ given as

$$D = \{x \in \mathbb{R}^2 \mid d(x, c) \leq \lambda\}$$

for some $c \in \mathbb{R}^2$ and $\lambda > 0$. The interior of D is an open set given as $\{x \in \mathbb{R}^2 \mid d(x, c) < \lambda\}$. The following property is known as a folklore (see, e.g., [5, Theorem 9.6]).

Proposition 1. *In the Delaunay graph $G = (P, E)$ of P, two nodes $u, v \in P$ are adjacent to each other if and only if there exists a closed disk $D \subseteq \mathbb{R}^2$ such that u and v lie on the boundary of D and any other nodes are not contained in the interior of D.*

For a pair (u, v) of distinct nodes of P, the (unique) disk $D(u, v)$ that has the line segment connecting u and v as its diameter chord is called the *Gabriel disk* of (u, v). If the Gabriel disk $D(u, v)$ contains no node in P other than u and v, the pair (u, v) is called an *Gabriel edge*. The graph on the vertex set P with Gabriel edges is called the *Gabriel graph* on P. Proposition 1 implies that every Gabriel edge is an edge in the Delaunay graph on P; that is, the Gabriel graph on P is a subgraph of a Delaunay graph. It is well known that a Gabriel graph is connected since it contains the Euclidean minimum spanning tree of the node set P as its subgraph.

3 Greedy Routing Using General Objective Functions

Denote by T the set of triplets of distinct nodes, i.e.,

$$T = \{(w, v, t) \mid w, v, t \text{ are distinct nodes in } P\}.$$

It is noted that each element $(w, v, t) \in T$ is an ordered set, i.e., (w, v, t) and (v, t, w) are different elements.

We propose a greedy routing algorithm using a general objective function $f : T \to \mathbb{R} \cup \{+\infty\}$. This routing algorithm repeatedly forwards a packet to some neighbor w of a current node v which minimizes the function value $f(w, v, t)$ among all neighbors of v until a packet reaches the destination node t. More precisely, the routing algorithm is described as follows.

Algorithm Generalized Greedy Routing
Step 0: Set $v := s$, where s is a given source node.
Step 1: If the destination node t is a neighbor of v in G, then set $v := t$ (i.e., forward a packet to t) and stop.

Step 2: Select a neighbor w of v in G which minimizes the value $f(w, v, t)$ among all neighbors of v, and set $v := w$ (i.e., forward a packet to w). Go to Step 1.

In particular, we consider restricted classes of objective functions in Generalized Greedy Routing. We say that an objective function $f : T \to \mathbb{R} \cup \{+\infty\}$ is *congruence-invariant* if the function value $f(w, v, t)$ depends only on the shape and the size of the triangle $\triangle wvt$ given by three nodes w, v, and t. More precisely, a function $f : T \to \mathbb{R} \cup \{+\infty\}$ is said to be congruence-invariant if there exists a function $h : \mathbb{R}^6_+ \to \mathbb{R} \cup \{+\infty\}$ such that

$$f(w, v, t) = h(d_{vt}, d_{wt}, d_{vw}, a_t, a_w, a_v) \qquad ((w, v, t) \in T),$$

where each parameter in h is given as follows:

$$d_{vt} = d(v, t),\ d_{wt} = d(w, t),\ d_{vw} = d(v, w),\ a_t = \angle vtw,\ a_w = \angle twv,\ a_v = \angle wvt.$$

It may be noted that the parameters $d_{vt}, d_{wt}, d_{vw}, a_t, a_w, a_v$ used in the function h are dependent; for example, $a_t + a_w + a_v = \pi$ holds. We keep this redundancy for simplicity of presentation. It is noted that the function values of a congruence-invariant objective function do not change even if we transform the given node set by a combination of translations, rotations, and reflections, i.e., the two node sets before and after the transformation are congruent.

In the following, we mainly discuss the algorithm Generalized Greedy Routing with a congruence-invariant objective function. Many existing greedy routing algorithms can be represented as a special case of Generalized Greedy Routing by using appropriate congruence-invariant objective functions, as shown below. Figure 1 shows contour maps of these congruence-invariant objective functions.

Greedy Routing [3]: It chooses a neighbor w with the minimum distance $d(w, t)$. The corresponding congruence-invariant function is given as

$$f_{\mathrm{G}}(w, v, t) = h_{\mathrm{G}}(d_{vt}, d_{wt}, d_{vw}, a_t, a_w, a_v) = d_{wt}.$$

Compass Routing [4]: It chooses a neighbor w with the minimum angle $\angle tvw$. The corresponding congruence-invariant function is given as

$$f_{\mathrm{C}}(w, v, t) = h_{\mathrm{C}}(d_{vt}, d_{wt}, d_{vw}, a_t, a_w, a_v) = a_v.$$

Midpoint Routing [7]: It chooses a neighbor w with the minimum distance between w and $(v + t)/2$, the midpoint of v and t. The corresponding congruence-invariant function is given as

$$\begin{aligned} f_{\mathrm{MP}}(w, v, t) &= h_{\mathrm{MP}}(d_{vt}, d_{wt}, d_{vw}, a_t, a_w, a_v) \\ &= (d_{wt} \sin a_t)^2 + (d_{wt} \cos a_t - (1/2)d_{vt})^2. \end{aligned}$$

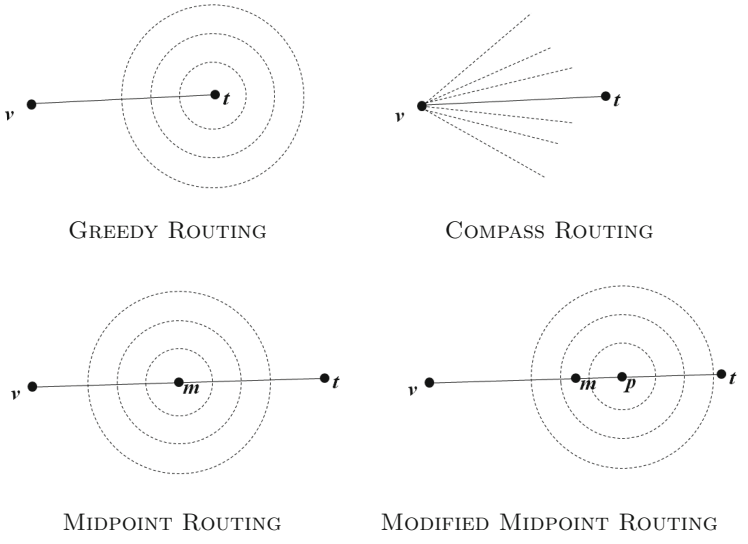


Fig. 1. Contour maps of congruence-invariant functions

Modified Midpoint Routing [7]: This algorithm is considered in [7] as a generalization of Midpoint Routing. Given a fixed real number $\lambda \in [0.5, 1]$, it chooses a neighbor w with the minimum distance between w and $(1-\lambda)v + \lambda t$. If $\lambda = 1$ (resp., $\lambda = 1/2$), then Modified Midpoint Routing coincides with Greedy Routing (resp., Midpoint Routing). The corresponding congruence-invariant function is given as

$$\begin{aligned} f_{\mathrm{MMP}}(w, v, t) &= h_{\mathrm{MMP}}(d_{vt}, d_{wt}, d_{vw}, a_t, a_w, a_v) \\ &= (d_{wt} \sin a_t)^2 + (d_{wt} \cos a_t - \lambda d_{vt})^2. \end{aligned}$$

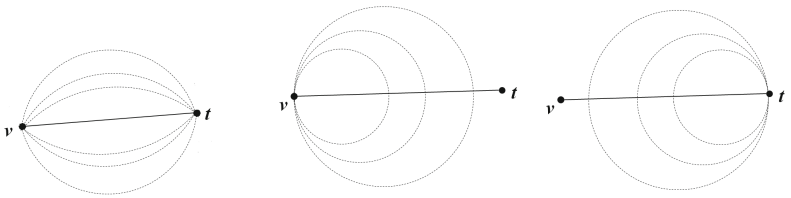
In addition, we propose three new greedy routing algorithms, each of which can be represented as a special case of Generalized Greedy Routing by using some congruence-invariant objective functions. For $\alpha \in \mathbb{R}$, let $\varphi_\alpha : \mathbb{R} \to \{0, +\infty\}$ be a function given by

$$\varphi_\alpha(\beta) = \begin{cases} 0 & (\text{if } \beta < \alpha), \\ +\infty & (\text{otherwise}). \end{cases}$$

New Greedy I: This is a variant of Compass Routing, and chooses a neighbor w with the *maximum* angle $\angle vwt$. The corresponding congruence-invariant function is given as

$$f_1(w, v, t) = h_1(d_{vt}, d_{wt}, d_{vw}, a_t, a_w, a_v) = -a_w.$$

New Greedy II: This can be seen as a combination of Compass Routing and Greedy Routing, and it chooses a neighbor w with the minimum value of



New Greedy I New Greedy II New Greedy III

Fig. 2. Contour maps of new congruence-invariant functions

$d(v,w)/\cos(\angle tvw)$ under the condition that $\angle tvw < \pi/2$. The corresponding congruence-invariant function is given as

$$f_2(w,v,t) = h_2(d_{vt}, d_{wt}, d_{vw}, a_t, a_w, a_v) = \frac{d_{vw}}{\cos a_v} + \varphi_{\pi/2}(a_v).$$

New Greedy III: This can be also seen as a combination of Compass Routing and Greedy Routing, and it chooses a neighbor w with the minimum value of $d(w,t)/cos(\angle wtv)$ under the condition that $\angle wtv < \pi/2$. The corresponding congruence-invariant function is given as

$$f_3(w,v,t) = h_3(d_{vt}, d_{wt}, d_{vw}, a_t, a_w, a_v) = \frac{d_{wt}}{\cos a_t} + \varphi_{\pi/2}(a_t).$$

Figure 2 shows contour maps of congruence-invariant functions of algorithms to give visual intuition of ideas. Each of the proposed algorithm has an advantage over the previously proposed algorithms, as explained in Section 5.

4 Guaranteed Delivery on Delaunay Graphs

As mentioned in Introduction, all of the four algorithms Greedy Routing, Compass Routing, Midpoint Routing, and Modified Midpoint Routing, each of which is a special case of Generalized Greedy Routing, always guarantee packet delivery on Delaunay graphs. In this section, we derive a sufficient condition for Generalized Greedy Routing to guarantee packet delivery on Delaunay graphs.

Lemma 1. *Let G be the Delaunay graph of P and $u, v \in P$ be distinct nodes of G. Suppose that u and v lie on the boundary of some closed disk $D \subseteq \mathbb{R}^2$ and u is not a neighbor of v in G. Then, there exists a neighbor w of v in G such that w is in the interior of D.*

Proof. By Proposition 1, there exists some $w \in P$ contained in the interior of D since u and v are not adjacent to each other. Let $w_1, w_2, \ldots, w_k \in P$ be the nodes contained in the interior of D. For each $i = 1, 2, \ldots, k$, let D_i be the closed disk such that w_i and v lie on the boundary of D_i and (the boundary of) D_i is

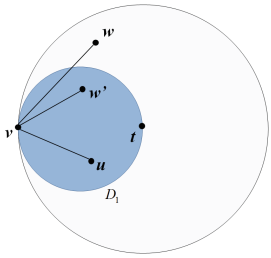


Fig. 3. DDG condition

tangent to (the boundary of) D at v. Assume, without loss of generality, that the radius of D_i is smaller than or equal to that of D_{i+1} for $i = 1, 2, \ldots, k-1$. Then, D_1 does not contain any nodes of $P \setminus \{v, w_1\}$ in its interior. Hence, w_1 is a neighbor of v in G by Proposition 1. □

We consider the following Delaunay Delivery Guarantee (DDG) condition for an objective function f (see Figure 3).

> **(DDG)** For every distinct nodes $w, v, t \in P$, if there exists some $u \in D(v,t)$ such that $f(u,v,t) \geq f(w,v,t)$, then $d(w,t) < d(v,t)$ holds.

In Figure 3, the dark disk is the Gabriel disk $D_1 = D(v,t)$ of (v,t), and the condition $d(w,t) < d(v,t)$ means that w is in the larger disk around t. Thus, DDG condition can be rephrased as follows: if the contour curve of f through a node w intersects the interior of the Gabriel disk of a current node v and the destination node t, then w is in the larger disk.

We also consider a stronger version of the DDG condition:

> **(SDDG)** For every distinct nodes $w, v, t \in P$, the inequality $f(w,v,t) > \max\{f(u,v,t) \mid u \in P \setminus D(v,t)\}$ holds.

It is easy to see that the strong DDG condition implies the DDG condition; if the strong DDG condition holds, then there exists no $u \in D(v,t)$ satisfying the condition in (DDG). The strong DDG condition intuitively means that the contour curve of f grows in $D(v,t)$ and eventually coincides with the boundary circle of $D(v,t)$, and then continues expansion to the outside of $D(v,t)$.

The next theorem shows that this condition implies the guaranteed delivery of the algorithm Generalized Greedy Routing on Delaunay graphs.

Theorem 1. *Let $f : T \to \mathbb{R} \cup \{+\infty\}$ be a function satisfying the* DDG *condition. Then, the algorithm* Generalized Greedy Routing *with objective function f guarantees packet delivery on a graph G if it is a supergraph of the Delaunay graph of P.*

Proof. If t is adjacent to v, the algorithm certainly delivers the packet. Suppose that v is a current node which is not adjacent to the destination node t, and

w is the neighbor node chosen by the algorithm. To prove that the algorithm guarantees packet delivery, it suffices to show the inequality $d(w,t) < d(v,t)$ since there exist a finite number of nodes in P.

Since v and t are non-adjacent nodes in G, they are non-adjacent in the Delaunay graph. Lemma 1 implies that there exists a neighbor u of v in G such that $u \in D(v,t)$. By the choice of w, we have $f(u,v,t) \geq f(w,v,t)$. Hence, the DDG condition implies $d(w,t) < d(v,t)$. □

We then show that the objective functions of Greedy Routing, Midpoint Routing, and Modified Midpoint Routing satisfy the DDG condition. This fact provides an alternative proof for the guaranteed delivery on Delaunay graphs.

Lemma 2. *$f_{\mathrm{G}}, f_{\mathrm{MP}}$, and f_{MMP} satisfy the* DDG *condition.*

Proof. Since f_{G} and f_{MP} are special cases of f_{MMP} with $\lambda = 0$ and $\lambda = 1/2$, respectively, we consider the function f_{MMP} only. Below we omit the subscript of f_{MMP} for simplicity.

Let (w,v,t) be an element in T such that w and v are adjacent. Suppose that

$$f(w,v,t) \leq \max\{f(u,v,t) \mid u \in P,\ (u,v,t) \in T,\ u \in D(v,t)\} \tag{1}$$

We will show that $d(w,t) < d(v,t)$ holds.

Let $p = (1-\lambda)v + \lambda t$. Since $f(w,v,t) = d(w,p)$, the inequality (1) can be rewritten as

$$d(w,p) \leq \max\{d(u,p) \mid u \in P,\ (u,v,t) \in T,\ u \in D(v,t)\}.$$

It holds that

$$\begin{aligned} &\max\{d(u,p) \mid u \in P,\ (u,v,t) \in T,\ u \in D(v,t)\} \\ &< \sup\{d(x,p) \mid x \in \mathbb{R}^2,\ d(x,c) \leq d(v,c)\} \ = \ d(v,p). \end{aligned}$$

Hence, we have $d(w,p) < d(v,p)$. This implies that

$$d(w,t) \leq d(w,p) + d(p,t) < d(v,p) + d(p,t) = d(v,t).$$

□

Note that among the three algorithms above, only Midpont Routing satisfies the strong DDG condition.

Theorem 2. *The algorithms* Greedy Routing, Midpoint Routing, *and* Modified Midpoint Routing *guarantee packet delivery on Delaunay graphs and their supergraphs.*

Proof. The statement follows immediately from Theorem 1 and Lemma 2. □

We next show that our new routing algorithms are also supported by Delaunay graphs. The following lemma immediately follows from the fact that contour curves grows inside the Gabriel disk of (v,t), which is observed easily from Figure 2.

Lemma 3. *Functions f_1, f_2 and f_3 satisfy the strong* DDG *condition.*

Combined with Theorem 1, this lemma implies the following:

Theorem 3. *The algorithms* New Greedy I, II, *and* III *guarantee packet delivery on Delaunay graphs and their supergraphs.*

Finally, we show that the class of objective functions satisfying the (strong) DDG condition is closed under convex combination, where a convex combination means a linear combination with nonnegative coefficients. This implies that the above mentioned routing algorithms can be used as basis of space of a class of the generalized greedy routing algorithms.

Theorem 4. *For every objective functions f and g satisfying the* DDG *(resp., strong* DDG*) condition, their convex combination also satisfies the* DDG *(resp., strong* DDG*) condition.*

Proof. We consider the case of the DDG condition only since the case of the stronger DDG condition is easier to prove. Consider a convex combination $F = \lambda f + (1 - \lambda)g$ of f and g, where $0 \leq \lambda \leq 1$. Let $w, v, t \in P$ be any distinct nodes such that $d(w,t) \geq d(v,t)$, and we show that $\max\{F(u,v,t) \mid u \in D(v,t)\} < F(w,v,t)$ holds. Since f and g satisfy the DDG condition, we have $\max\{f(u,v,t) \mid u \in D(v,t)\} < f(w,v,t)$ and $\max\{g(u,v,t) \mid u \in D(v,t)\} < g(w,v,t)$, from which $\max\{F(u,v,t) \mid u \in D(v,t)\} < F(w,v,t)$ follows. □

5 Comparison of Routing Algorithms

Listed below are two important properties required for geometric routing algorithms:

- Long-edge avoidance: long edges (i.e. a communication using large transmission radius) should not be used as far as possible.
- Fast transmission: each packet should be sent with a small number of hops.

We investigate the special cases of the generalized greedy routing algorithms considered in this paper from the viewpoint of the two properties, which are in general incompatible requirements.

We observed that Midpoint Routing, New Greedy I, New Greedy II, and New Greedy III satisfy the strong DDG condition. This means that if a graph G contains Delaunay graph as a subgraph, the routing algorithm always finds a neighbor in the Gabriel disk of the current node and the destination node. This is advantageous since there is no possibility that a path becomes highly zigzag. Other algorithms do not enjoy this property since the contour curves may properly intersect the boundary of the Gabriel disk.

Let us examine algorithms one by one. In order to reduce the number of hops in the network, it is advantageous to reduce the distance to the destination node as much as possible. For this purpose, the algorithm Greedy Routing

is apparently the best. It, however tends to select long edges, and also possibly visit a node outside the Gabriel disk as we remarked above.

Midpoint Routing selects shorter edges than Greedy Routing, while keeping the number of hops to be small (intuitively, it is about the twice of that of Greedy Routing). It also always selects a node in the Gabriel disk of the current node and the destination node. It, however, still tends to select a long edge since its most favorite neighbor location is the midpoint whose edge length is the half of the distance $d(v, t)$. Modified Midpoint Routing uses a parameter to control this balance, while it uses longer edge than Midpoint Routing because of constraint that $\lambda \geq 1/2$. Note that if we choose $\lambda < 1/2$, the resulting objective function violates the DDG condition, and packet delivery is not guaranteed.

By definition New Greedy I is designed to use as straight path as possible since a contour curve closer to the line segment vt has a smaller value of the objective function f. As a side effect, the total length of the path is expected to be short in most of cases. It, however, is difficult to control the length of edges in a path.

New Greedy II and New Greedy III have contour curves which are circles going through v and t, respectively. In this sense, it resembles to Modified Midpoint Routing, while they enjoy additional nice properties. Intuitively, New Greedy II tends to select shorter edges, while New Greedy III selects long edges with smaller visual angles than Modified Midpoint Routing. More specifically, if the graph G contains a Delaunay graph, New Greedy II always selects a Delaunay edge (an edge in a Delaunay graph) as its next hop since the contour curve with the minimum value of $f(w, v, t)$ is an empty disk that has v and w on its boundary. Thus, we need not examine which edge is a Delaunay edge in G since the algorithm automatically selects such a Delaunay edge. This is advantageous since we enjoy properties of Delaunay graphs without identifying Delaunay edges explicitly. Using the same logic, we can show that if G contains a Delaunay graph and there is a node w that is a common neighbor of v and t in the Delaunay graph, then New Greedy III selects a neighbor of t (it may not be w). Consequently, if there exists a two-hop path from v to t in the Delaunay graph, the algorithm also finds such a path.

In summary, we should use an appropriate method depending on the requirement of an ad hoc network. In this sense, our proposed algorithms are useful since they enjoy good properties that seems to be desired in many occasions. Since the DDG condition is closed under convex combination as mentioned in the previous section, we can design hybrid of algorithms in response to the request of users.

6 Concluding Remarks

We compared several geometric routing algorithms, and gave a unified view on a family of algorithms. Based on a basic property called DDG condition, we propose several variants of algorithms jthat also work on a Delaunay graph. Although we considered the problem on the Euclidean plane, its extension to

the three dimensional space can be naturally considered. In the real ad hoc network design, we often need to consider a metric space with nonuniform distance because of existence of obstacles and other natural/social conditions. Handling such metric spaces is a challenging topic as a future work.

Acknowledegment. This work is partially supported by a Grant-in-Aid for Scientific Research from the Ministry of Education, Culture, Sports, Science and Technology of Japan. The authors thank anonymous referees of ALGOSENSORS 2012 for their comments which are useful for the revision of the paper.

References

1. Bose, P., Morin, P.: Online routing in triangulations. SIAM J. Comput. 33, 937–951 (2004)
2. Bose, P., Devroye, L., Loeffler, M., Snoeyink, J., Verma, V.: The spanning ratio of the Delaunay triangulation is greater than $\pi/2$. In: Proc. CCCG 2009, pp. 165–167 (2009)
3. Finn, G.G.: Routing and addressing problems in large metropolitan-scale internetworks. University of Southern California, Tech. Rep. ISI/RR-87-180 (1987)
4. Kranakis, E., Singh, H., Urrutia, J.: Compass routing on geometric networks. In: Proc. CCCG 1999, pp. 51–54 (1999)
5. de Berg, M., van Kreveld, M., Overmars, M., Schwarzkopf, O.: Computational Geometry, Algorithms and Applications, 2nd edn. Springer, Berlin (1999)
6. Rührup, S.: Theory and practice of geometric routing. In: Liu, H., Chu, X., Leung, Y.-W. (eds.) Ad Hoc and Sensor Wireless Networks: Architectures, Algorithms and Protocols. Bentham Science (2009)
7. Si, W., Zomaya, A.Y.: Midpoint routing algorithms for Delaunay graphs. In: Proc. IPDPS 2010, pp. 1–7 (2010)
8. Stojmenovic, I.: Position based routing in ad hoc networks. IEEE Commun. Magazine 40, 128–134 (2002)
9. Takagi, H., Kleinrock, L.: Optimal transmission ranges for randomly distributed packet radio terminals. IEEE Trans. Commun. 32, 246–257 (1984)

Big Data Interpolation an Efficient Sampling Alternative for Sensor Data Aggregation (Extended Abstract)*

Hadassa Daltrophe, Shlomi Dolev, and Zvi Lotker

Ben-Gurion University, Beer-Sheva, 84105, Israel
{hd,dolev}@cs.bgu.ac.il,
zvilo@bgu.ac.il

Abstract. Given a large set of measurement sensor data, in order to identify a simple function that captures the essence of the data gathered by the sensors, we suggest representing the data by (spatial) functions, in particular by polynomials. Given a (sampled) set of values, we interpolate the datapoints to define a polynomial that would represent the data. The interpolation is challenging, since in practice the data can be noisy and even Byzantine, where the Byzantine data represents an adversarial value that is not limited to being close to the correct measured data. We present two solutions, one that extends the Welch-Berlekamp technique in the case of multidimensional data, and copes with discrete noise and Byzantine data, and the other based on Arora and Khot techniques, extending them in the case of multidimensional noisy and Byzantine data.

Keywords: Big Data, Sensor Data Aggregation, Sampling, Data Interpolation, Spatial Sensor Inputs.

1 Introduction

Consider the task of representing information in an error-tolerant way, such that it can be introduced even if it contains noise or even if the data is partially corrupted and destroyed. Polynomials are a common venue for such approximation, where the goal is to find a polynomial p of degree at most d that would represent the entire data correctly.

* Partially supported by a Russian Israeli grant from the Israeli Ministry of Science and Technology #85387301-"Algorithmic approaches to energy savings" and the Russian Foundation for Basic Research, the Rita Altura Trust Chair in Computer Sciences, the Lynne and William Frankel Center for Computer Sciences, Israel Science Foundation (grant number 428/11), Cabarnit Cyber Security MAGNET Consortium, Grant from the Institute for Future Defense Technologies Research named for the Medvedi of the Technion, MAFAT, and Israeli Internet Association.

A. Bar-Noy and M.M. Halldórsson (Eds.): ALGOSENSORS 2012, LNCS 7718, pp. 66–77, 2013.

Our motivation comes from sensor data aggregation, and the need to extend the distributed aggregation to distributed interpolation, use sampling to cope with huge data and anticipate the value of missing data. For example, a sensor network may interact with the physical environment, while each node in the network is may sense the surrounding environment (e.g., temperature, humidity etc). The environmental measured values should be transmitted to a remote repository or remote server. Note that the environmental values usually contain noise, and there can be malicious inputs, i.e., part of the data may be corrupted.

In contrast to distributed data aggregation where the resulting computation is a function such as COUNT, SUM and AVERAGE (e.g. [16,13]),in distributed data interpolation, our goal is to represent every value of the data by a single (abstracting) function. Our computational model consists of sampling the sensor network data and estimating the missing information using polynomial manipulations.

The management of big data systems also gives motivation for the distributed interpolation method. The abstraction of big data becomes one of the most important tasks in the presence of the enormous amount of data produced these days. Communicating and analyzing the entire data does not scale, even when data aggregation techniques are used. This study suggests a method to represent the distributing big data by a simple abstract function (such as polynomial) which will lead to effective use of that data.

We suggest interpolating the big data in the scope of distributed systems by using local *data centers*. Each data center samples the data around it and computes a polynomial that reflects the local data. The local polynomials are merged to a global one by interpolation in a hierarchical manner. In the process of calculating the local polynomials noise and Byzantine data samples are eliminated.

Basic Definitions

- For *multivariate polynomial* $p(\mathtt{x}) \in \mathbb{R}[\mathtt{x}] = \mathbb{R}[x_1, ..., x_k]$ let

$$\|p\|_\infty = sup\{|p(x_1, ..., x_k)| : x_1, ..., x_k \in \mathbb{R}\}$$

- A *monomial* in a collection of variable $x_i, ..., x_n$ is a product $x_1^{\alpha_1} x_2^{\alpha_2} x_n^{\alpha_n}$ where α_i are non-negative integers.
- The *total degree* of a multivariate polynomial p is the maximum degree of any term in p, where the degree of particular term is the sum of the variable exponents.
- A polynomial q is a δ-*approximation* to p if $\|p - q\|_\infty \leq \delta$.

Polynomial Fitting to Noisy and Byzantine Data

Formally, in this paper, we learn the following problem:

Definition 1 (Polynomial Fitting to Noisy and Byzantine Data Problem). *Given a sample S of k dimension datapoints $\{(x_{1_i}, ..., x_{k_i})\}_{i=1}^N$ and a*

function f defined on those points $f(x_{1_i}, ..., x_{k_i}) = y_i$, a noise parameter $\delta > 0$ and a threshold $\rho > 0$, we have to find a polynomial p of total degree d satisfying

$$p(x_1, ..., x_k) \in [y - \delta, y + \delta] \text{ for at least } \rho \text{ fraction of } S \tag{1}$$

Generally, we propose the use of polynomials to represent large amounts of sensor data. The process works by sampling the data and then using this sample to construct a polynomial whose distance (according to the ℓ_∞ metric) from the polynomial constructed using the *whole* data set is small. The main challenges to this approach are (i) the presence of noise (identified by the δ parameter), and (ii) arbitrarily corrupted data (Byzantine data, denoted by ρ) that can cause inaccurate sampling and, thus, lead to badly constructed polynomials.

Given that the function f is continuous, by the Weierstrass approximation Theorem [4] we know that for any given $\epsilon > 0$, there exists a polynomial p' such that

$$\|f - p'\|_\infty < \epsilon \tag{2}$$

This can tell us that our desired polynomial p exists (i.e., $p' = p$ and $\epsilon = \delta$, satisfying eq.1), and we can relate the data as arising from polynomial function (i.e., the unknown function f is d degree polynomial we need to reconstruct), and this is the underlying model assumed in the paper.

One obvious candidate to construct approximating polynomial is interpolation at equidistant points. However, the sequence of interpolation polynomials does *not* converge uniformly to f for all $f \in C[0,1]$ due to Runge's phenomenon [7]. Chebyshev interpolation (i.e., interpolate f using the points defined by the Chebyshev polynomial) minimizes Runge's oscillation, but it is not suffice the polynomial fitting problem presented above (Definition 1) due to the randomly distributed data we have assumed.

Taylor polynomials are also not appropriate; for even setting aside questions of convergence, they are applicable only to functions that are *infinitely differentiable*, and not to all continuous functions.

Another classical polynomial sequence is suggested by S. Bernstein [3] as constructive proof of the Weierstrass Theorem. Bernstein polynomial: $B_n^f(x) = \sum_{i=0}^{n} f\left(\frac{i}{n}\right)\binom{n}{i}x^i(1-x)^{n-i}$ converges uniformly to any continuous function f which is bounded on $[0,1]$. The formal Berenstein polynomial samples the function f in an *equidistant* fashion. To handle a random sample data, we can use Vitale [21] results which consider that the datapoints $S = x_1, ..., x_N$ are *i.i.d* observations drawn from an unknown *density* function f. The *Bernstein polynomial estimate of* f defined as $\tilde{B}_n^f(x) = \frac{n+1}{N}\sum_{i=0}^{n}\mu_{in}^N\binom{n}{i}x^i(1-x)^{n-i}$ where μ_{in}^N is the number of points (x_i's) appear in the interval $[\frac{i}{n+1}, \frac{i+1}{n+1}]$. Vitale [21] showed that $\left\|\tilde{B}_n^f(x) - f\right\|_\infty \leq \epsilon$ for every given $\epsilon > 0$.

Tenbusch [20] extended Vitale's idea to multidimensional densities, where there is need to note that those works hold only when the datapoints are *i.i.d*

observations. Another reason not to use the Bernstien polynomial is the slow convergence rate (Voronovskaya's Theorem states that for functions that are twice differentiable, the rate of convergence is about $1/n$, see Davis [7]).

Considering other classical curve-fitting and approximation theories [17], most research has used the ℓ_2 norm of noise, such as the method of least square errors. These attitudes not suffice the adversarial noise we have assumed here. To our knowledge, only [2] referred the ℓ_∞ noise that fits our considered problem and we further relate [2] study.

The polynomial fitting problem as stated in Definition 1 can also be studied by Error-Correcting Code Theory. From that point of view, extensive literature exists dealing with the noise-free case (i.e., $\delta = 0$ and $\rho < 1$). In the next section, we present an algorithm that handles a combination of discrete noise and Byzantine data based on the Welch-Berlekamp [22] error-elimination method. Moreover, the fundamental Welch-Berlekamp algorithm treats only the one-dimension case, where we suggest a means to deal with corrupted-noisy data appearing at one and multi-dimensional inputs.

Related to unrestricted noise, we refer to the polynomial-fitting problem as defined by Arora and Khot [2]. Based on their results in Section 3, we introduce the polynomial fit generalization, where we provide a polynomial-time algorithm dealing with multi-variate data.

Summarizing, this work provides the following contributions:

- We describe an algorithm that constructs a polynomial using the Welch-Berlekamp (WB) method as a subroutine. The algorithm is tolerated to discrete-noise and Byzantine data.
- We identify how the previous method can be generalized to handle multi-dimensional data. Moreover, we present a multivariate analogue of the WB method, under conditions which will be specified.
- Using linear programing minimization and the Markov-Bernstein Theorem, we generalized Arora and Khot algorithm to reconstruct an unknown *multi-dimensional* polynomial. Furthermore, we detail the way to eliminate the Byzantine appearance when such inputs exist.

Those three points stated in the three algorithms presented in the paper. The first Algorithm handles one-dimensional Byzantine data that contains discrete-noise. Algorithm 2 generalized the WB idea to deal with multivariate malicious data. Finally, Algorithm 3 summarized our approach to cope with unrestricted noise appeared in the (partially corrupted) data. Note that this paper does not include the full theorems proofs'. The whole details can be found at [6].

2 Discrete Finite Noise

In this section, we will study a simple aspect of the polynomial fitting problem posed in Definition 1, where the data function is a polynomial, and we relaxed the noise constraint to be finite and discrete, i.e., the noise δ is defined on a finite field $\mathbb{F}_q$ containing q elements.

Welch and Berlekamp related the problem of polynomial reconstruction in their decoding algorithm for Reed Solomon codes [22]. The main idea of their algorithm is to describe the received (partially corrupted) data as a ratio of polynomials. Their solution holds for noise-free cases and a limited fraction of the corrupted data ($\delta = 0, \rho > 1/2$). Almost 30 years later, Sudan's list decoding algorithm [19] relaxed the Byzantine constraint ($\delta = 0, \rho$ can be less than $1/2$) by using bivariate polynomial interpolation. Those concepts do not hold up well in the noisy case since they use the roots of the polynomial and the divisibility of one polynomial by other methods that are problematic for noisy data (as shown in [2], Section 1.2). Here, we will use the WB algorithm [22] as a "black box" to obtain an algorithm that handles the discrete-noise notation of the polynomial-fitting problem.

Given a data set $\{(x_i, y_i)\}_{i=1}^{N}$ that is within a distance of $t = \rho N$ from some polynomial $p(x)$ of degree $< d$, the WB approach to eliminate the irrelevant data is to use the roots of an object called *the error-locating polynomial*, e. In other words, we want $e(x_i) = 0$ whenever $p(x_i) \neq y_i$. This is done by defining another polynomial $q(x) \equiv p(x)e(x)$. To resolve these polynomials we need to solve the linear system, $q(x_i) = y_i e(x_i)$ for all i.

Welch and Berlekamp show that $e(x)|q(x)$ and $p(x)$ can be found by the ratio $p(x) = q(x)/e(x)$ at $O(N^\omega)$ running time (where ω is the matrix multiplication complexity). In Algorithm 1, we are use the WB method as a subroutine to manage the noisy-corrupted data.

Algorithm 1. Reconstruct the polynomial $p(x)$ representing the true data

Require: $S, S' \subseteq S, \rho, d, \delta, \Delta = v_1, ..., v_{|S'|}$
  $i \leftarrow 0$
  repeat
    $i \leftarrow i + 1$
    $S'_i \leftarrow S' + v_i$
    $p_i(x) \leftarrow WB(S'_i, d, \rho)$
  until $p_i(x_j) \in [y_j - \delta, y_j + \delta]$ for at least ρ fraction of j's; $(x_j, y_j) \in S - S'_i$
  return $p(x) \leftarrow p_i(x)$

Given *any* sample S such that ρ fraction of S is not corrupted, we will choose a subset $S' \subseteq S$ in a size related to the desired degree d and ρ (the WB algorithm requires $2t + d$ points, where $t = \rho N$ is the number of the corrupted points). At every step i, we will add S' different values of noise as defined by the set Δ which contain all the vectors of length $|S'|$ assigned the elements of $\mathbb{F}_q$ in lexicographic order, i.e., $\Delta = \left\{(a_1, ..., a_{|S'|}) : a_i \in \mathbb{F}_q\right\}$. Now, we can reconstruct the polynomial p_i using the WB algorithm. The resulting polynomial p_i is tested by the original dataset S, where the criteria is that p_i is within δ from all nodes but the Byzantine nodes (according to the maximal number of Byzantine as defined by ρ).

Since we assume a discrete finite noise ($\delta \in \mathbb{F}_q$), for each datapoint at the subset S' (of size $O(d + \rho|S|)$), there is a possibly of q values (where q is a

constant). Thus, in the worst case, when we run the WB polynomial algorithm for every possible value, it will cost $poly(d+N)$ time.

Note that if the desired polynomial's degree d is not given, we can search for the minimal degree of a polynomial that fits the δ and number of Byzantine node restrictions in a binary search fashion.

Multidimensional Data

To generalize the former algorithm to handle multidimensional data, there is need to formalize the WB algorithm to deal with multivariate polynomials. This is a challenging task due to the *infinite* roots those polynomials may have (and as previously mentioned, the WB method is strictly based on the polynomials' roots).

A suggested method to handle 3-dimensional data is to assume that the values of datapoints in one direction (e.g., x-direction) are distinct. This can be achieved by assuming the inputs $S = (x_1, y_1, f(x_1, y_1))..., (x_N, y_N, f(x_N, y_N))$ are *i.i.d* observations. Moreover, we allow the malicious authority to change the observation input but not its distribution (i.e., to determine $z_i = f(x_i, y_i)$ value only). This assumption forces the data to have different x_i's values, which help us to define the error locating polynomial e in the x-direction only (or symmetrically over the y-axis).

The 3-dimensional polynomial reconstruction is described in Algorithm 2.

Algorithm 2. Reconstruct the polynomial $p(x, y)$ representing the true data

- **Input**: $0 < t = \rho N$ which is the Byzantine appearance bound, the total degree $d > 1$ of the goal polynomial and N triples $(x_i, y_i, z_i)_{i=1}^{N}$ with distinct x_i's .
- **Output**: Polynomial $p(x, y)$ of total degree at most d or fail.
- **Step 1**: Compute a non-zero univariate polynomial $e(x)$ of degree exactly t and a bivariate polynomial $q(x, y)$ of total degree $d + t$ such that:

$$z_i e(x_i) = q(x_i, y_i) \; 1 \leq i \leq N \tag{3}$$

  If such polynomials do not exist, output fail.
- **Step 2**: If e does not divide q, output fail, else compute $p(x, y) = \frac{q(x,y)}{e(x)}$. If $\Delta(z_i, p(x_i, y_i)_i) > t$, output fail. else output $p(x, y)$.

Theorem 1. *Let p be an unknown d total degree polynomial with two variables. Given a threshold $\rho > 0$ and a sample S of $N = \binom{d+t+m}{d+m} + t$ $(t = \rho N)$ random points $(x_i, y_i, z_i)_{i=1}^{N}$ such that*

$$z_i = p(x_i, y_i) \text{ for at least } \rho \text{ fraction of } S.$$

The algorithm above reconstructs p at $O(N^\omega)$ running time (where ω is the matrix multiplication complexity).

Proof. The proof of the Theorem above follows from the subsequent claims.

Claim (Correctness). There exist a pair of polynomials $e(x)$ and $q(x,y)$ that satisfy **Step 1** such that $q(x,y) = p(x,y)e(x)$.

Proof. Taking the error locator polynomial $e(x)$ and $q(x,y) = p(x,y)e(x)$, where $deg(q) \leq deg(p) + deg(e) \leq t + d$. By definition, $e(x)$ is a degree t polynomial with the following property:

$$e(x) = 0 \text{ iff } z_i \neq p(x,y)$$

We now argue that $e(x)$ and $q(x,y)$ satisfy eq. 3. Note that if $e(x_i) = 0$, then $q(x_i,y_i) = z_i e(x_i) = 0$. When $e(x_i) \neq 0$, we know $p(x_i,y_i) = z_i$ and so we still have $p(x_i,y_i)e(x_i) = z_i e(x_i)$, as desired.

Claim (Uniqueness). If any two distinct solutions $(q_1(x,y), e_1(x)) \neq (q_2(x,y),$ $e_2(x))$ satisfy **Step 1**, thenbthey will satisfy $\frac{q_1(x,y)}{e_1(x)} = \frac{q_2(x,y)}{e_2(x)}$.

Proof. It suffices us to prove that $q_1(x,y)e_2(x) = (q_2(x,y)e_1(x)$. Multiply this with z_i and substitute x, y with x_i, y_i, respectively,

$$q_1(x_i,y_i)e_2(x_i)z_i = q_2(x_i,y_i)e_1(x_i)z_i$$

We know, $\forall i \in [N]$ $q_1(x_i,y_i) = e_1(x_i)z_i$ and $q_2(x_i,y_i) = e_2(x_i)z_i$ If $z_i = 0$, then we are done. Otherwise, if $z_i \neq 0$, then $q_1(x_i,y_i) = 0, q_(x_i,y_i) = 0 \Rightarrow$ $q_1(x,y)e_2(x) = (q_2(x,y)e_1(x)$ as desired.

Claim (Time complexity). Given $N = t + \binom{d+t+2}{d+t}$ data samples, we can reconstruct $p(x,y)$ using $O(N^\omega)$ running time.

Proof. Generally, for m variate polynomial with degree d, there are $\binom{d+m}{d}$ terms [18]; thus, it is a necessary condition that we have $t+\binom{d+t+2}{d+t}$ distinct points for q and e to be uniquely defined. We have N linear equation in at most N variables, which we can solve e.g., by Gaussian elimination in time $O(N^\omega)$ (where ω is the matrix multiplication complexity).

Finally, **Step 2** can be implemented in time $O(NlogN)$ by long division [1]. Note that the general problem of deciding whether one multivariate polynomial divides another is related to computational algebraic geometry (specifically, this can be done using the Gröbner base). However, since the divider is a univariate polynomial, we can mimic long division, where we consider x to be the "variable" and y to just be some "number."

Example 1. Suppose the unknown polynomial is $p(x,y) = x + y$. Given the parameters: $d = 1$ (degree of p), $m = 2$ (number of variable at p) and $t = 1$ (number of corrupted inputs) and the set of $t + \binom{d+t+2}{d+t} = 7$ points:

(1,2,2),(2,2,4),(6,1,7),(4,3,7),(8,2,0),(9,1,10),(3,7,10)

that lie on $z = p(x, y)$. Following the algorithm, we define: $deg(e) = 1, deg(q) = 2$ and

$$q_i = \alpha_1 x_i^2 + \alpha_2 x_i y_i + \alpha_3 y_i^2 + \alpha_4 x_i + \alpha_5 y_i + \alpha_6 = z_i(x_i + \alpha_7)$$

for coefficients $\alpha_1, ..., \alpha_6, \beta$ and $1 \leq i \leq 12$. Note that we force $e(x)$ not to be the zero polynomial by define it to be monic (i.e., the leading coefficient equals to 1). Thus, we derive the linear system:

$$\begin{aligned}
\alpha_1 + \alpha_2 + \alpha_3 + \alpha_4 + \alpha_5 + \alpha_6 &= 2\beta + 2 \\
4\alpha_1 + 4\alpha_2 + 4\alpha_3 + 2\alpha_4 + 2\alpha_5 + \alpha_6 &= 4\beta + 8 \\
36\alpha_1 + 6\alpha_2 + \alpha_3 + 6\alpha_4 + \alpha_5 + \alpha_6 &= 7\beta + 42 \\
16\alpha_1 + 12\alpha_2 + 9\alpha_3 + 4\alpha_4 + 3\alpha_5 + \alpha_6 &= 7\beta + 28 \\
64\alpha_1 + 16\alpha_2 + 4\alpha_3 + 8\alpha_4 + 2\alpha_5 + \alpha_6 &= 0 \\
81\alpha_1 + 9\alpha_2 + \alpha_3 + 9\alpha_4 + \alpha_5 + \alpha_6 &= 10\beta + 90 \\
9\alpha_1 + 21\alpha_2 + 49\alpha_3 + 3\alpha_4 + 7\alpha_5 + \alpha_6 &= 10\beta + 30
\end{aligned}$$

Solving the system gives: $q(x, y) = x^2 + xy - 8x - 8y$ and $e(x) = x - 8$. Dividing those polynomials, we get the expected solution: $q(x, y)/e(x) = p(x) = x + y$.

Corollary 1 (Multivariate Polynomial Reconstruction). *Let p be an unknown d total degree polynomial with m variable. Given a threshold $\rho > 0$, a noise parameter δ and a sample S of N random points $(x_{1_i}, ..., x_{m_i}, y_i)_{i=1}^{N}$ such that*

$$y_i \in [p(x_{1_i}, ..., x_{m_i}) - \delta, p(x_{1_i}, ..., x_{m_i}) + \delta] \text{ for at least } \rho \text{ fraction of } S$$

p can be reconstructed using $N = \binom{d+m+\rho m}{d+m} + \rho m$ datapoints.

3 Random Sample with Unrestricted Noise

Motivated by applications in vision, Arora and Khot [2] studied the univariate polynomial fitting to noisy data using $O(d^2)$ datapoints, where d is the polynomial degree. In this part, we generalized their results to k-dimensional data.

Since our motivation comes from sensor planar aggregation, we will focus on bivariate polynomial reconstruction, where the multivariate proof is symmetric. We assume by rescaling the data that each $x_i, y_i, f(x_i, y_i) \in [-1, 1]$. Allowing small noise at every point and large noise occasionally then there may be too many polynomials agreeing with the given data. Thus, given the noise parameter δ, our goal is to find a polynomial p that is a δ-approximation of f, i.e., p is δ-close in ℓ_∞ norm to the unknown polynomial.

Let I be a set of d^5 equally spaced points that cover the interval $[-1, 1]$. Given the random sample $S \subset I, |S| = \frac{d^2}{\delta} log(\frac{d}{\delta})$, we approach the reconstruction problem by defining a linear programming system with the fitting polynomial as its solution. To incorporate the constraint that the unknown polynomial must take values of $[-1, 1]$, we move to Chebyshev's representation of the polynomial.

Thus, each of its coefficients is at most $\sqrt{2}$ (see eq. 5). We represent Chebyshev's polynomial by $T_i(\cdot), T_j(\cdot)$, and the variables c_{ij} at the system are the Chebyshev coefficients. We construct the LP:

$$\text{minimize } \delta \text{ s.t.}$$

$$f(x_k, y_k) - \delta \leq \sum_{i}^{n}\sum_{j}^{m} c_{ij} T_i(x_k) T_j(y_k) \leq f(x_k, y_k) + \delta,\ k = 1, ..., |S| \quad (4)$$

$$|c_{ij}| \leq \sqrt{2},\ i = 1, ..., n; j = 1, ..., m \quad (5)$$

$$|\sum_{i}^{n}\sum_{j}^{m} c_{ij} T_i(x) T_j(y)| \leq 1, \forall_{x,y} \in I \quad (6)$$

The following Theorem presents our main result for solving the polynomial fitting problem:

Theorem 2. *Let f be an unknown d total degree polynomial with two variables, such that $f(x, y) \in [-1, 1]$ when $x, y \in [-1, 1]$. Given a noise parameter $\delta > 0$, a threshold $\rho > 0$, a constant $c > 0$ (dependent on the dimension of the data) and a sample S of $O(\frac{d^2}{\delta} log(\frac{d}{\delta}))$ random points $x_i, y_i, z_i \in [-1, 1]$ such that $z_i \in [f(x_i, y_i) - \delta, f(x_i, y_i) + \delta]$ for at least ρ fraction of S. With probability at least $\frac{1}{2}$ (over the choice of S), any feasible solution p to the above LP is $c\delta$-approximation of f.*

Remarks:

- If we know that the derivative is bounded by Δ (i.e., $f'_x, f'_y \leq \Delta$), the above proof that gives us $\frac{\Delta}{\delta} log \frac{\Delta}{\delta}$ points is sufficient.
- The Bernstein-Markov Theorem which Theorem 2 based on also holds for multivariate trigonometric polynomials (see [9]), thus, we can generalize the above proof also for this class of function. This generalization is important in the scope of wireless sensor networks since the use of trigonometric function is the appropriate way to represent the sensor data behavior (e.g., temperature).
- The presented method holds only when we assume equidistance or random sampling (as opposed to Section 2 that handles any given sample). Otherwise, when the dataset is dense, since we allow δ perturbation of the data, it can cause a sharp slope in the resulting function although the original data is close to the constant at the sampling interval.

Corollary 2. *Given the set S of $O(\frac{d^2}{\delta} log(\frac{d}{\delta}))$ k-dimensional random datapoints and a constant $c(S, k)$ dependent only on the geometry and the dimension of the data, we can reconstruct the unknown polynomial within $c(S, k)\delta$ error in ℓ_∞ norm with high probability over the choice of the sample.*

Proof. The two-dimensional proof holds for the general dimension, where the approximation accuracy dependent on the constant $c(S,k)$ comes from Bernstein-Markov Theorem. This constant is independent of the polynomial degree, but dependent on the set of the data points (see [9]). Note that $c(S,k)$ increases exponentially when increasing the dimension.

Byzantine Elimination

Arora and Khot [2] do not deal with Byzantine inputs; however, the method they presented in Section 6 can be rewritten to eliminate corrupted data such that the input datapoints will contain only true (but noisy) values.

Assume that ρ fraction of the data is uncorrupted. For any point $x_i, y_i \in [-1,1]$, consider a small square-interval $\Lambda = [x_i - \frac{\delta}{d^3}, x_i + \frac{\delta}{d^3}] \times [y_i - \frac{\delta}{d^3}, y_i + \frac{\delta}{d^3}]$ (where d is the total degree of the polynomial we need to find). For a sample of $d^4 \frac{log(1/\delta)}{\delta}$ points, with high probability $\Omega(log(d))$ of the samples lie in this square. We are given that ρ fraction of these sample points gives an approximate value of $f(x_i, y_i)$, i.e., the correct value lies in the interval $[f(x_i, y_i) - \delta, f(x_i, y_i) + \delta]$ and the rest of the sample is corrupted and, thus, is NOT in $[f(x_i, y_i) - \delta, f(x_i, y_i) + \delta]$. Following Bernstein-Markov Theorem, the derivatives are bounded by $O(d^2)$; thus, the value of the polynomial is essentially *constant* over Λ. Hence, at least ρ fraction of the values seen in this square will lie in $[f(x_i, y_i) - \delta, f(x_i, y_i) + \delta]$ and the rest is irrelevant corrupted data. Thus, at every point (x_i, y_i), we can reconstruct $f(x_i, y_i)$. The sample is large enough so that we can reconstruct the values of the polynomial at say, d^2/δ equally spaced points. Now, applying the techniques presented in Section 3 enables us to recover the polynomial.

Reconstructing the Multivariate Polynomial

To conclude this section, we summarize the presented results in Algorithm 3:

Algorithm 3. Reconstruct the polynomial $p(x,y)$ representing the true data

Require: S, ρ, d, δ
  $S' \leftarrow \emptyset$
  $i \leftarrow 1$
  repeat
    $\Lambda = [x_i - \frac{\delta}{d^3}, x_i + \frac{\delta}{d^3}] \times [y_i - \frac{\delta}{d^3}, y_i + \frac{\delta}{d^3}]$
    $c \leftarrow \frac{z_1 + \ldots + z_k}{k}, z_j : (x_j, y_j) \in \Lambda$
    $S' \leftarrow \{(x_j, y_j, z_j) | (x_j, y_j) \in \Lambda \wedge z_j \approx c\}$
    $i \leftarrow i + 1$
  until $|S'| > \frac{d^2}{\delta}$
  $p(x,y) \leftarrow$LP minimization (Equations 4-6) on the set S'
  return $p(x,y)$

The algorithm requires the dataset S, the true-data fraction ρ, the total degree of the expected polynomial d and the noise parameter δ. In the first phase, we eliminate the Byzantine occurrence, as described in the former subsection. Assuming the given data lie in $[-1,1] \times [-1,1]$ (or translate to that interval), for the points in S, we are looking at the $\frac{\delta}{d^3}$-close interval and choose all the points that have *constant* value at this interval (this is done by the average operation). We repeat this process until we collect enough true-datapoints, i.e., at least $\frac{d^2}{\delta}$ points. This set (sign as S' in the algorithm) is the input for the linear-programming equations which finally give us the expected polynomial as proof at Theorem 2.

4 Conclusions

We have presented the concept of data interpolation in the scope of sensor data aggregation and representation, as well as the new big data challenge, where abstraction of the data is essential in order to understand the semantics and usefulness of the data. Interestingly, we found that classical techniques used in numeric analysis and function approximation, such as the Welsh-Berlekamp efficient removal of corrupted data, Arora Khot and the like, relate to the data interpolation problem. Since the sensor aggregation task is usually a collection of inputs from *spatial* sensors, for the first time we have extended existing classical techniques for the case of three or even more function dimensions, finding polynomials that approximate the data in the presence of noise and limited portion of completely corrupted data.

We believe that the mathematical techniques we have presented have applications beyond the scope of sensor data collection or big data, in addition to being an interesting problem that lies between the fields of error-correcting and the classical theory of approximation and curve fitting.

Throughout the research we have distinguished two different measures for the polynomial fitting to the Byzantine noisy data problem: the first being the Welsh-Berlekamp generalization for discrete-noise multidimensional data and the second being the linear-programming evaluation for multivariate polynomials.

Approached by the error-correcting code methods, we have suggested a way to represent a noisy-malicious input with a multivariate polynomial. This method assumes that the noise is discrete. When the noise is unrestricted, based on Bernstein-Markov Theorem and Arora & Khot algorithm, we have suggested a method to reconstruct algebraic or trigonometric polynomial that traverses ρ fraction of the the noisy multidimensional data.

We suggest to use polynomial to represent the abstract data since polynomial is dense in the function space on bounded domains (i.e., they can approximate other functions arbitrarily well) and have a simple and compact representation as oppose to spline e.g., [11] or others image processing methods.

Directions for further investigation might include the use of interval computation for representing the noisy data with interval polynomials.

References

1. Aho, A.V., Hopcroft, J.E., Ullman, J.D.: The Design and Analysis of Computer Algorithms, vol. 8. Addison-Wesley Publishing Company (1974)
2. Arora, S., Khot, S.: Fitting algebraic curves to noisy data. In: STOC, pp. 162–169 (2002)
3. Bernstein, S.: Demonstration du theoreme de Weierstrass, fondee sur le calcul des probabilities. Communications of the Kharkov Mathematical Society 2(13), 1–2 (1912-1913)
4. Bishop, E.: A generalization of the Stone Weierstrass theorem. Pacific Journal of Mathematics 11(3), 777–783 (1961)
5. Clenshow, C.W., Hayes, J.G.: Curve and Surface Fitting. J. Inst. Maths Applics. 1, 164–183 (1965)
6. Daltrophe, H., Dolev, S., Lotker, Z.: Data Interpolation, An Efficient Sampling Alternative for Big Data Aggregation. The Lynne and William Frankel Center of Computer Science of Ben-Gurion University of the Negev, Technical Report 13-01 (September 2012)
7. Davis, P.J.: Interpolation and approximation. Dover (1975)
8. de Boor, C., Ron, A.: Computational aspects of polynomial interpolation in several variables. Math. Comp. 58, 705–727 (1992)
9. Ditzian, Z.: Multivariate Bernstein and Markov inequalities. Journal of Approximation Theory 70(3), 273–283 (1992)
10. Faugére, J.C.: A new efficient algorithm for computing Gröbner base. Journal of Pure and Applied Algebra 139(1-3), 61–88 (1999)
11. Guenther, N.: Approximation by spline functions. Springer, Berlin (1989)
12. Hilbert, M., Lepez, P.: The World's Technological Capacity to Store, Communicate, and Compute Information. Science 332(6025), 60–65 (2011)
13. Jesus, P., Baquero, C., Almeida, P.S.: A Survey of Distributed Data Aggregation Algorithms. CoRR, abs/1110.0725 (2011)
14. Kingsley, E.H.: Bernstein polynomials for functions of two variables of class $C^{(k)}$. Proceedings of the American Mathematical Society, 64–71 (1951)
15. Lynch, C.: How do your data grow? Nature 455, 28–29 (2008)
16. Rajagopalan, R., Varshney, P.K.: Data aggregation techniques in sensor networks: a survey. IEEE Commun. Surveys Tutorials 8(4) (2006)
17. Rivlin, T.J.: An introduction to the approximation of function. Blaisdell publishing company (1969)
18. Saniee, K.: A Simple Expression for Multivariate Lagrange Interpolation. SIAM Undergraduate Research Online 1(1) (2008)
19. Sudan, M.: Decoding of Reed Solomon codes beyond the error-correction bound. Journal of Complexity 13(1), 180–193 (1997)
20. Tenbusch, A.: Two-dimensional Bernstein polynomial density estimators. Metrika 41(1), 233–253 (1994)
21. Vitale, R.A.: A Bernstein polynomial approach to density estimation. Statistical Inference and Related Topics 2, 87–100 (1975)
22. Welch, L.R., Berlekamp, E.R.: Error correction for algebraic block codes. US Patent 4 633 470 (1986)

Mapping a Polygon with Holes Using a Compass

Yann Disser[1], Subir Kumar Ghosh[2], Matúš Mihalák[1], and Peter Widmayer[1]

[1] Institute of Theoretical Computer Science, ETH Zurich, Switzerland
[2] School of Computer Science, Tata Institute of Fundamental Research, India

Abstract. We consider a simple robot inside a polygon $\mathcal{P}$ with holes. The robot can move between vertices of $\mathcal{P}$ along lines of sight. When sitting at a vertex, the robot observes the vertices visible from its current location, and it can use a compass to measure the angle of the boundary of $\mathcal{P}$ towards north. The robot initially only knows an upper bound $\bar{n}$ on the total number of vertices of $\mathcal{P}$. We study the mapping problem in which the robot needs to infer the visibility graph G_{vis} of $\mathcal{P}$ and needs to localize itself within G_{vis}. We show that the robot can always solve this mapping problem. To do this, we show that the minimum base graph of G_{vis} is identical to G_{vis} itself. This proves that the robot can solve the mapping problem, since knowing an upper bound on the number of vertices was previously shown to suffice for computing G_{vis}.

Keywords: autonomous robot, polygon, hole, mapping, exploration, visibility graph, compass, algorithm.

1 Introduction

The *mapping problem* and the *localization problem* are fundamental for many tasks in robotics and autonomous exploration. In the mapping problem, a robot is required to obtain a (rough) map of an initially unknown environment, while the localization problem requires the robot to identify its current position on the map. Both problems often arise together and need to be solved simultaneously. In this paper, we use the term *mapping problem* loosely to refer to the combination of both tasks.

The difficulty of mapping depends on the type of the environment as well as on the capabilities of the robot. Many variations in scenario, robot models, and questions have been studied in this context. We are interested in the following question: *What are minimal capabilities that a robot needs in order to solve the mapping problem?*

We study the mapping problem in polygonal environments. In particular, and in contrast to past work, we allow polygonal obstacles (or, equivalently, holes) in the environment. Our robot model is based on a minimalistic framework: Our *basic robot* can move from vertex to vertex along lines of sight, and, being at a vertex, the robot can observe other visible vertices in counterclockwise order. Other than that, the robot has no direct means of distinguishing vertices according to global identifiers or names, i.e., it can not even tell whether it has

A. Bar-Noy and M.M. Halldórsson (Eds.): ALGOSENSORS 2012, LNCS 7718, pp. 78–89, 2013.

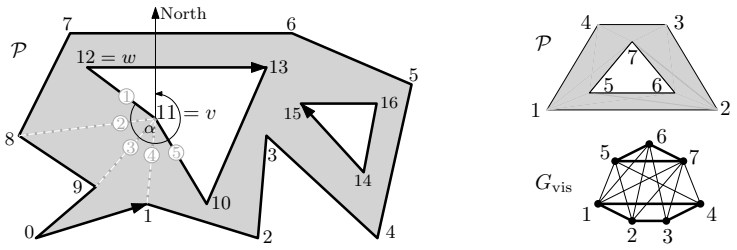


Fig. 1. Left: A robot located at vertex v of polygon $\mathcal{P}$ and its visible vertices (connected with grey line-segments) ordered as they appear in a counterclockwise scan of $\mathcal{P}$, starting on the boundary. Angle α is the angle between the boundary at v and north. Right: A polygon $\mathcal{P}$ and its visibility graph G_{vis}.

visited its current location before. Figure 1 illustrates the capabilities of the basic robot in a polygon with holes (for a formal definition, see Section 1.1). Using this model as a baseline we can compare different ways of equipping it with additional sensors (e.g., sensors that measure angles, distances, etc.). Our goal is to find the smallest set of extra capabilities that empowers the robot to solve the mapping problem. In the basic model, the robot obviously cannot hope to infer the geometry of the polygon it is exploring. Instead, we concentrate on reconstructing the *visibility graph* as a topological map of the polygon. The visibility graph of a polygon $\mathcal{P}$ is the graph $G_{\text{vis}} = (V, E)$, where V are the vertices of $\mathcal{P}$ and E contains the edge $\{u, v\}$ if and only if u and v see each other in $\mathcal{P}$ (i.e., the line segment connecting them does not leave $\mathcal{P}$). Figure 1 gives an example of a polygon with its visibility graph.

Suri et al. showed that a robot with a *pebble* can solve the mapping problem in polygons with holes, even without information about the size of the polygon [19]. Such a *pebble* is a way for the robot to mark a vertex: The robot can drop the pebble at its current location, it can distinguish the vertex that holds the pebble as long as it is visible, and it can pick the pebble back up later. A pebble is a powerful tool for the robot. It has been shown for example that a much weaker pebble that cannot be sensed from a distance allows a robot exploring any directed graph to reconstruct the graph [2]. It is an important question whether a robot with weaker abilities can solve the mapping problem in polygons with holes. To the best of our knowledge, no such result is known.

Without a pebble, the presence of holes makes mapping substantially more difficult. For example, consider the robot model introduced in [6]. There, the robot is equipped with the ability to *look-back*, i.e., the robot can identify the vertex from which it arrived in its last move, among its visible vertices. Using such a model, it was then shown that the robot can compute the visibility graph of any *simple* polygon (i.e., without holes), provided that a bound on the number of vertices is known. Figure 2 illustrates that the robot cannot infer the visibility graph in general if the polygon may have holes. In each of the three polygons in the example, the robot senses exactly the same, no matter how it moves. Therefore, there is no way it can distinguish the polygons. Moreover, the example

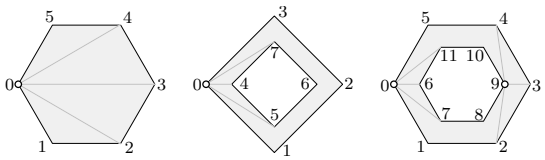


Fig. 2. Polygons that a robot with look-back cannot distinguish, even if it has an upper bound on the number of vertices. At every vertex of the three polygons, the robot observes exactly the same – including the information about which vertex it arrived from in its last move.

highlights other limitations: the robot cannot infer the number of vertices, and the robot cannot tell whether it is located on a hole. The example relies on the fact that the robot does not know the number of vertices exactly.

Figure 2 also illustrates an important structural property of simple polygons, which polygons with holes do not admit: A simple polygon always has an *ear*, i.e., a vertex whose neighbors on the boundary see each other. In the first polygon in Figure 2 every vertex is an ear, the other two polygons have no ears at all. This property is crucial for existing mapping techniques, because it allows an inductive approach based on "cutting off" ears repeatedly [6,7]. For polygons with holes, we cannot hope to make use of ears in similar fashion. Solving the mapping problem may thus require a more capable robot.

In this paper we consider the following extension to the basic robot model: the robot knows an upper-bound $\bar{n}$ on the number of vertices, and the robot has a *boundary compass*. A boundary compass allows to measure the angle at the robot's location formed by the line of sight to the counterclockwise neighbor along the boundary and towards *north*, where *north* is any global reference direction in the plane. Figure 1 illustrates the concept of a boundary compass. We show that such a robot can reconstruct the visibility graph of any polygon with or without holes.

Related Work. Various approaches have been made to modeling minimalistic robots for various environments and objectives [1,13,17,19]. Some effort has been devoted to classifying the power of robot models and to comparing different models in that respect [5,12,18]. The basic robot model that serves as a foundation in this paper was introduced in [19], and has been studied in [3,5,7,6,14].

The focus of research regarding the mapping problem in polygonal environments has so far been on *simple* polygons (i.e., without holes). We provide a brief overview over the different extensions of the basic robot model that have been studied in the past in the context of simple polygons. For a detailed discussion, we refer to [9].

It has been shown that the basic model does not always allow to infer the visibility graph of a polygon [5]. Such a robot can in general not even infer the number of vertices n. On the positive site, it has been shown that a robot can compute the visibility graph with the following extensions to the basic model: (i) the robot has a pebble [19]; (ii) the robot knows an upper bound $\bar{n}$ on the

number of vertices, and it has *look-back* [6]; (iii) the robot knows an upper bound $\bar{n}$ on the number of vertices, and it can tell convex from reflex angles, i.e., it can tell for any two visible vertices whether the angle between these two vertices is greater or smaller than π [7]; (iv) the robot has look-back and it can tell convex from reflex angles [3].

An even more minimalistic version of the basic robot model has been considered, the version that restricts the robot to moving along the boundary only. In this model it was shown that knowing the number of vertices n is not sufficient to reconstruct the visibility graph, even when the robot can measure the angle formed by the boundary at each vertex [3]. If the robot can measure the angles between any two lines of sight, however, reconstruction is possible even without prior knowledge of n [8,11,10].

An inherent difficulty of visibility graph reconstruction is that these graphs have not yet successfully been characterized [15,16].

1.1 Problem Definition

Polygon. Throughout this paper we consider the exploration of a polygon $\mathcal{P}$ with polygonal holes and a total of n vertices. We write $\mathcal{H}_1, \mathcal{H}_2, , \mathcal{H}_h$ to denote the holes of $\mathcal{P}$ and $\bar{\mathcal{P}}$ for the enclosing polygon of $\mathcal{P}$ without holes. The *boundaries* of $\mathcal{P}$ consist of the boundary of $\bar{\mathcal{P}}$ together with the boundaries of $\mathcal{H}_1, \mathcal{H}_2, \ldots, \mathcal{H}_h$. We will sometimes refer to the boundary of $\bar{\mathcal{P}}$ as the *outer boundary*.

Two vertices u and v are *mutually visible* in $\mathcal{P}$ (or *see each other*) if the line segment connecting the two vertices does not leave $\mathcal{P}$. We call the line-segment between u and v the *line of sight* (between the two vertices).

The *counterclockwise* neighbor of vertex v of $\mathcal{P}$ is v's neighbor u on the boundary of $\mathcal{P}$ such that by moving along the line segment from u to v, the interior of $\mathcal{P}$ lies to the left of the line segment. Figure 1 illustrates this by the arrows on the boundary suggesting the order of vertices which we encounter if we move (iteratively) to the counterclockwise neighbor. Observe the difference of the order if placed on a hole or on the outer boundary of $\mathcal{P}$: an iterative process of moving to the counterclockwise neighbor results in a (a) a counterclockwise walk if the robot moves on the outer boundary $\bar{\mathcal{P}}$, or (b) a clockwise walk if the robot moves on a hole.

Robot. A robot is modeled as a "moving point". Initially, the robot is placed at a vertex of $\mathcal{P}$. Being at a vertex v, the robot observes the following information about $\mathcal{P}$: (i) the number of vertices visible to v, and (ii) the angle $\alpha_v^{\uparrow}$ of the ray from v to its counterclockwise neighbor on the boundary towards a globally fixed direction that we will refer to as *north*. See Figure 1 for illustration.

The robot can order the visible vertices: starting on the boundary, the robot can sort all lines of sight at v as they appear in a counterclockwise scan of the polygon. This naturally induces an ordering of the visible vertices (Figure 1 illustrates this ordering). The robot can select a position in this ordering and move to the corresponding vertex (without knowing the global identity of it).

Visibility Graph. The visibility graph of a polygon $\mathcal{P}$ is an undirected graph $G_{\text{vis}} = (V, E)$, where V is the set of vertices of $\mathcal{P}$, and E contains the edge $\{u, v\}$ if and only if u and v see each other in $\mathcal{P}$ (i.e., edges of G_{vis} are lines of sight of $\mathcal{P}$). To reflect the local sensing of a robot at a vertex $u \in V$, we will consider a directed and edge-labeled version of the visibility graph. We replace every undirected edge $\{u, v\}$ with two directed edges (u, v) and (v, u). We label every edge (u, v) by a label $l(u, v)$ which encodes the information observed by a robot at vertex u. Formally, we set $l(u, v) = (i, \alpha_u^{\uparrow})$, where i denotes that (u, v) is the i-th line of sight at u in counterclockwise order, and $\alpha_u^{\uparrow}$ is the angle between the ray towards u's counterclockwise neighbor and north. Observe that the edge (u, v) will generally have a different label than the edge (v, u). With this transformation, we can regard the robot operating inside the polygon $\mathcal{P}$ as an agent moving along the edges of the directed and edge-labeled visibility graph G_{vis}, where the agent sees the labels of the outgoing edges of the vertex it is located at.

Minimum Base Graph. An edge-labeled directed multi-graph G' is a *base graph* of an edge-labeled directed graph G, if every vertex v of G can be mapped to a vertex v' of G' such that every path in G starting at v with an induced sequence λ of edge labels has a corresponding path starting at v' in G' with the same induced sequence λ of edge labels, and vice versa, i.e., every path in G' has a counterpart in G. A *minimum* base graph G^* of G is a base graph of G of minimum size. Every graph G has a unique minimum base graph G^* (up to isomorphism) [4].

A useful interpretation of the minimum base graph is to see every vertex of the minimum base graph G^* as representing a class of vertices of G. Every two vertices of G that map to the same vertex of G^* belong to the same class. Each class groups vertices together according to the same observation along paths in G specified by a sequence of edge-labels (recall that at any vertex u there are no two adjacent outgoing edges with the same label, and thus any sequence of edge-labels uniquely specifies a path in G). This is useful when arguing about the robot: Starting from any vertex in a class, the robot observes the same for every sequence of movement decisions. In other words, the vertices of the same class are indistinguishable by the robot by means of moving and sensing. Moreover, a minimum base graph G^* can be used as a kind of map as well. Being located in vertex $v \in G_{\text{vis}}$ and knowing the corresponding class v^* in G^*, we can use G^* to navigate the robot to any other class of G^*.

For an example, consider the visibility graphs in Figure 2. For the basic robot, i.e., without sensing the angles $\alpha_v^{\uparrow}$, the edge-label of every edge in the directed visibility graph only encodes the position of the corresponding line of sight in the local ordering. In that case, it is easy to observe that the multi-graph consisting of one node with five self-loops labeled (1), (2), (3), (4), and (5), is indistinguishable from the three visibility graphs by the robot. Because there is obviously no smaller such graph, it is the *minimum* base graph of each of the three visibility graphs. Obviously, computing the minimum base graph does not help the basic

robot to solve the mapping problem: as far as it can tell, it could be in any of the three polygons – the minimum base graph does not help to distinguish them.

In this paper we will show that if the robot also has a boundary compass (i.e., it can measure $\alpha_v^{\uparrow}$), then the minimum base graph of every correspondingly edge-labeled visibility graph G_{vis} is the visibility graph itself. Therefore, computing the minimum base graph is enough to compute G_{vis}.

Goal. We want to know whether the robot can infer the visibility graph of any polygon $\mathcal{P}$ and determine its location in it. More precisely, given a number $\bar{n} \geq n$, we want to know whether there exists a deterministic algorithm that (i) navigates the robot inside any polygon $\mathcal{P}$ with at most $\bar{n}$ vertices, and (ii) computes from the collected observations the visibility graph G_{vis} of $\mathcal{P}$, as well as the robot's location in G_{vis}.

2 Algorithm

In this section we show that a robot with boundary compass can compute the visibility graph G_{vis} of any polygon $\mathcal{P}$ if it knows an upper bound $\bar{n}$ on the number of vertices of $\mathcal{P}$. We do so by showing that the minimum base graph G^* is equal to the visibility graph G_{vis}. Using the algorithm of [7,9] for determining the minimum base graph, the algorithm then trivially follows: Since $G^* = G_{\text{vis}}$, we can simply apply the algorithm of [7,9] and return its result. The algorithm operates on general edge-labeled graphs and also determines the location of the robot in G^* (and thus, in our case, in G_{vis} as well). We note that the running time of the algorithm can be exponential in $\bar{n}$ in the worst case.

We can see this approach as a generic black-box method for solving the mapping problem by some variant of the basic robot (with extended sensing capabilities), assuming that an upper bound $\bar{n}$ on the number of vertices is known. The method is as follows, with its core difficulty lying in step 2.

1. Encode sensed information in the edge-labels of the directed version of G_{vis};
2. Show that $G^* = G_{\text{vis}}$;
3. Compute G^* using the algorithm of [7,9].

2.1 Labeling the Visibility Graph

We consider the directed and edge-labeled version of G_{vis} as described in Section 1.1. This labelling reflects the local sensing of the robot. Recall that every outgoing edge (u, v) of a vertex u is labeled with $(i, \alpha_u^{\uparrow})$, where i denotes the rank of v in the counterclockwise order of the vertices visible to u, and $\alpha_u^{\uparrow}$ is the angle formed by the ray to the counterclockwise neighbor of u along the boundary and north.

Because of the ordering of the lines of sight, no two labels of outgoing edges at a vertex are the same. Therefore, any walk in the visibility graph can uniquely be described by a starting vertex and a sequence of edge labels.

2.2 Showing That $G^* = G_{\text{vis}}$

To show that the minimum base graph G^* is equal to the visibility graph G_{vis}, we show that every two vertices u and v of G_{vis} are distinguishable by a walk in G_{vis} (thus showing that the two vertices cannot be in the same class of G^*). We proceed in several steps. In the following, we let $\mathcal{H} \in \{\bar{\mathcal{P}}, \mathcal{H}_1, \mathcal{H}_2, \ldots, \mathcal{H}_h\}$ be a hole $\mathcal{H}_i$ or the enclosing polygon $\bar{\mathcal{P}}$.

How to Distinguish Two Vertices of $\mathcal{H}$. We will show that no two vertices of $\mathcal{H}$ belong to the same class of G^*, i.e., we show that every two vertices of $\mathcal{H}$ are distinguishable by the robot.

The robot can consciously walk along the boundary of $\mathcal{H}$: It can just repeatedly move to its counterclockwise neighbor on the boundary (i.e., to its first visible vertex). This will result in a counterclockwise (if $\mathcal{H} = \bar{\mathcal{P}}$) or clockwise (if $\mathcal{H} \in \{\mathcal{H}_1, \mathcal{H}_2, \ldots, \mathcal{H}_h\}$) walk along the boundary of $\mathcal{H}$, in which the robot possibly visits each vertex of $\mathcal{H}$ more than once. Any such walk induces a sequence of observations (provided by the sensing capabilities of the robot). Let $n_{\mathcal{H}}$ denote the number of vertices of $\mathcal{H}$. After at most $\bar{n}$ steps, the robot is guaranteed to have visited every vertex of $\mathcal{H}$ at least once. Therefore, in a walk along the boundary of $\mathcal{H}$, observations repeat with a *period* of at most $n_{\mathcal{H}}$. Formally, $p \in \mathbb{N}$ is a period of a sequence $(a_i)_{i\in\mathbb{N}}$ if $a_i = a_{i+kp}$ for all $k \in \mathbb{N}$, and we say that the first p elements $(a_1, \ldots, a_p)$ *repeat* in the sequence. We show in the following that we can uniquely identify the exact value of $n_{\mathcal{H}}$ by considering the sequence of observations induced by $\bar{n}$ moves along the boundary.

We will use the following facts about the sum of the *inner* and *outer* angles, and about the *rotation number* of a simple polygon. Inner angles of $\mathcal{P}$ are the angles on the inside of $\mathcal{P}$ formed by two adjacent segments of a boundary. An outer angle is the counterpart of an inner angle – the angle on the outside of $\mathcal{P}$ formed by two adjacent boundary segments. See Figure 3 for illustration. The *rotation number* of a simple polygon measures (in angles), informally, how much the boundary turns. Formally, consider three consecutive neighbors u, v, w on the boundary of a simple polygon $\mathcal{P}$ in a chosen direction (counterclockwise or clockwise direction). The *turn angle* of the polygon at vertex v (in the chosen direction) is the angle at v formed by the rays $\overrightarrow{uv}$ and $\overrightarrow{vw}$ in this order (!). The rotation number in the chosen direction of $\mathcal{P}$ is the sum of its turn angles in the chosen direction. Turn angles are signed: a "left" turn gives a positive angle $\alpha_T \in (0, \pi)$, and a "right" turn gives a negative turn angle $\alpha_T \in (-\pi, 0)$. Figure 3 illustrates these angles in an example.

Fact 1. *The sum of all inner angles of a simple polygon is $(n-2)\pi$. The sum of all outer angles of a simple polygon is $(n+2)\pi$. The rotation number in the counterclockwise direction is 2π, and the rotation number in the clockwise direction is -2π.*

We use the *robot direction* to denote the direction which the robot induced on $\mathcal{H}$ if it walks along the boundary by iteratively moving to its counterclockwise

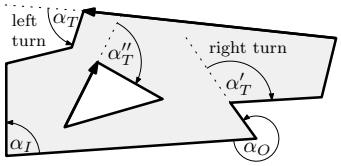


Fig. 3. A polygon with inner angle α_I, outer angle α_O and three turn angles $\alpha_T > 0$, $\alpha'_T < 0$, and $\alpha''_T < 0$. Observe that α_T and α'_T are the turn angles in counterclockwise direction, whereas α''_T is the turn angle in clockwise direction.

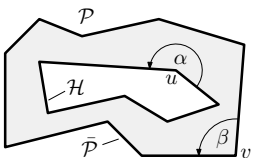


Fig. 4. Inner angles of polygon $\mathcal{P}$: α is the inner angle of the polygon at vertex of $u \in \mathcal{H}$ and β is the inner angle of the polygon at vertex $v \in \bar{\mathcal{P}}$. Observe that when $\mathcal{H}$ is considered as a simple polygon, then α is its outer angle.

neighbor. That is, if $\mathcal{H}$ is the outer boundary $\bar{\mathcal{P}}$, then *robot direction* is the counterclockwise direction, otherwise (if $\mathcal{H}$ is a hole $\mathcal{H}_i$) then *robot direction* is the clockwise direction.

We use the rotation number to infer $n_{\mathcal{H}}$ – the size of $\mathcal{H}$. While moving along the boundary of $\mathcal{H}$ the turn angles can be computed by using the angles provided by the boundary compass.

Proposition 1. *Let u be a vertex of $\mathcal{H}$ and let v be the counterclockwise neighbor of u on $\mathcal{H}$. Then, the turn angle in the robot direction is equal to $\alpha_u^{\uparrow} - \alpha_v^{\uparrow}$ (mapped to the interval $(-\pi, \pi)$).*

For every vertex $u \in \mathcal{H}$, the walk along the boundary induces a sequence $\boldsymbol{\alpha}(u) = (\alpha_1, \alpha_2, \ldots, \alpha_i, \ldots, \alpha_{\bar{n}}, \ldots)$ of turn angles. Recall that $\alpha_i \in (-\pi, \pi)$. By Fact 1 we know that $\sum_{i=1}^{n_{\mathcal{H}}} \alpha_i = \pm 2\pi$. For simplicity of the exposition, we will assume that the sum equals 2π. The case when the sum equals -2π can be handled analogously. Thus, we have $\sum_{i=1}^{k \cdot n_{\mathcal{H}}} \alpha_i = k \cdot 2\pi$. Obviously, the sequence $(\alpha_1, \ldots, \alpha_{n_{\mathcal{H}}})$ appears periodically in $\boldsymbol{\alpha}(u)$ with period $n_{\mathcal{H}}$. We claim the following.

Lemma 1. *Sequence $(\alpha_1, \ldots, \alpha_{n_{\mathcal{H}}})$ is the only sequence that periodically repeats in $\boldsymbol{\alpha}(u)$ and sums to 2π.*

Proof. Consider the sequence $(\alpha_1, \ldots, \alpha_k)$, $k \in \mathbb{N}$. We will show that if the sequence $(\alpha_1, \ldots, \alpha_k)$ sums to 2π and periodically repeats in $\boldsymbol{\alpha}(u)$, then $k = n_{\mathcal{H}}$. Assume therefore that the sequence repeats and sums to 2π, i.e., $\sum_{i=1}^{k} \alpha_i = 2\pi$. Consider the sum $X := \sum_{i=1}^{k \cdot n_{\mathcal{H}}} \alpha_i$. By the assumption, we can write the sum as $X = n_{\mathcal{H}} \sum_{i=1}^{k} \alpha_i = n_{\mathcal{H}} \cdot 2\pi$. At the same time, because $(\alpha_1, \ldots, \alpha_{n_{\mathcal{H}}})$ periodically repeats in $\boldsymbol{\alpha}(u)$, we also have $X = k \sum_{i=1}^{n_{\mathcal{H}}} \alpha_i = k \cdot 2\pi$. Therefore, $k = n_{\mathcal{H}}$. □

Lemma 1 immediately gives the robot a way to compute the number of vertices of $\mathcal{H}$. It suffices to identify the smallest period of $\boldsymbol{\alpha}(u)$ that sums to 2π. This is an easy task since the robot has an upper bound $\bar{n}$ on the total number of vertices – the robot walks $\bar{n}!$ number of steps along the boundary and identifies the smallest period in the resulting sequence $(\alpha_1, \ldots, \alpha_{\bar{n}!})$ that sums to 2π.

We can now show that every two vertices of $\mathcal{H}$ are distinguishable.

Lemma 2. *No two vertices of $\mathcal{H}$ belong to the same class of the minimum base graph G^* of G_{vis}.*

Proof. We have shown that, for any vertex $u \in \mathcal{H}$, the sequence $(\alpha_1, \dots, \alpha_{n_h})$ is the only sequence that periodically repeats in $\boldsymbol{\alpha}(u)$ and sums to 2π.

Consider any two vertices $u, v \in \mathcal{H}$. We claim that the walk along the boundary of size $n_{\mathcal{H}}$ distinguishes the two vertices. Obviously, if $\boldsymbol{\alpha}(u) \neq \boldsymbol{\alpha}(v)$, then also the corresponding sequences of observed angles $\angle^{\uparrow}_w$ at vertices w along the walk cannot be the same (as the latter implies the former). In the case when $\boldsymbol{\alpha}(u) = \boldsymbol{\alpha}(v)$ we have that the subsequence of $\boldsymbol{\alpha}(u)$ between u and v repeats in $\boldsymbol{\alpha}(u)$. Let p be the distance between u and v on the walk. Thus, p is the period of the subsequence in $\boldsymbol{\alpha}(u)$ and we have that p divides $n_{\mathcal{H}}$. Thus, the subsequence between u and v repeats $n_{\mathcal{H}}/p$ times within the first $n_{\mathcal{H}}$ elements of $\boldsymbol{\alpha}(u)$. It follows that $\sum_{i=1}^{p} \alpha_i =: \beta \neq 0$ (as otherwise $\sum_{i=1}^{n_{\mathcal{H}}} \alpha_i = 0$, a contradiction). Also, $\beta < 2\pi$ by Lemma 1. But then $\alpha^{\uparrow}_u \neq \alpha^{\uparrow}_v$ because $\alpha^{\uparrow}_v = \alpha^{\uparrow}_u - \beta$. Hence, obviously, the two vertices are distinguishable. □

How to Distinguish a Vertex of $\bar{\mathcal{P}}$ from the Rest. We now show that the robot can distinguish any vertex of $\bar{\mathcal{P}}$ from the vertices in $\mathcal{P} \setminus \bar{\mathcal{P}}$. Obviously, any vertex $u \in \bar{\mathcal{P}}$ can be distinguished from a vertex $v \in \mathcal{H}_i$, if $n_{\bar{\mathcal{P}}} \neq n_{\mathcal{H}_i}$: The walk along the boundary from the respective vertices u and v will induce sequences $\boldsymbol{\alpha}(u)$ and $\boldsymbol{\alpha}(v)$ of different periods. Let us therefore concentrate on the case where $n_{\bar{\mathcal{P}}} = n_{\mathcal{H}_i}$. We use Fact 1 again.

For Fact 1 we have defined an inner angle of a *simple* polygon. We can naturally define an *inner angle of polygon $\mathcal{P}$ with holes* to be the angle lying inside $\mathcal{P}$ and formed by two adjacent boundary segments. Note that for a vertex $v \in \mathcal{H}_i$, the corresponding inner angle in $\mathcal{P}$ is actually the outer angle of the simple polygon $\mathcal{H}_i$ (see Figure 4 for illustration).

Therefore, by Fact 1, the sum of the inner angles of $\mathcal{P}$ at vertices of $\bar{\mathcal{P}}$ is $(n_{\bar{\mathcal{P}}} - 2)\pi$, whereas the sum of the inner angles of $\mathcal{P}$ at vertices of $\mathcal{H}_i$ is $(n_{\mathcal{H}_i} + 2)\pi$.

Let $\mathcal{H}$ be a hole or $\bar{\mathcal{P}}$ at which the robot is positioned. The observed angles by the robot in a walk around the boundary of $\mathcal{H}$ induce the inner angles of $\mathcal{P}$ at vertices of $\mathcal{H}$. This then gives an immediate way to distinguish $\bar{\mathcal{P}}$ from any hole $\mathcal{H}_i$.

Proposition 2. *The inner angle of $\mathcal{P}$ at a vertex $v \in \mathcal{H}$ is equal to $\pi - \alpha_T$, where α_T is the signed turn angle at vertex v.*

Lemma 3. *No two vertices $u \in \bar{\mathcal{P}}$, $v \in \mathcal{H}_i$ belong to the same class of the minimum base graph G^*.*

Proof. We have already argued that the vertices are distinguishable if the number of vertices in $\bar{\mathcal{P}}$ and $\mathcal{H}_i$ is different. Without loss of generality, assume that $n_{\bar{\mathcal{P}}} = n_{\mathcal{H}_i} = n'$. By Fact 1 we know that the inner angles of $\bar{\mathcal{P}}$ sum up to $(n' - 2)\pi$ and the inner angles of any hole sum up to $(n' + 2)\pi$. Proposition 2 provides a

correspondence between the sequence of n' observed angles from the boundary compass and the inner angles of $\mathcal{P}$. As the sums of the inner angles are different for vertices of $\bar{\mathcal{P}}$ and vertices of a hole, the sequence of observed angles have to be different, too. Therefore, a counterclockwise walk along the boundary allows to distinguish between u and v. □

How to Distinguish Vertices of Different Holes. Recall that the vertices of $\bar{\mathcal{P}}$ can be uniquely distinguished in G^*, i.e., they form a singleton class in G^*. Therefore, as the robot can navigate in G^* to get to any class of G^*, it can get to any vertex of $\bar{\mathcal{P}}$.

We use the boundary of $\bar{\mathcal{P}}$ as a reference point to identify all other vertices of $\mathcal{P}$ uniquely. Consider an arbitrary vertex $v_1 \in \bar{\mathcal{P}}$. We use it as a kind of "origin" of G_{vis} to distinguish any two vertices $u \in \mathcal{H}_i$, $v \in \mathcal{H}_j$, $i \neq j$. Observe that, because G_{vis} is strongly connected, there exists a closed walk from v_1 that visits all vertices of G_{vis} – a Hamiltonian walk. We will now see the walk as a sequence L of both classes (vertices) of G^* and of edge-labels: every walk in G_{vis} translates to a walk in G^*; we add the visited vertices of G^* into L in the order induced by the walk. We will abuse the notation a bit, and use L to sometimes refer to the walk and sometimes to the edge-labels.

A sufficient condition to distinguish any two vertices u and v is that the walk L does not have a period smaller than $|L|$ (where $|L|$ denotes the length of the sequence). Having such a walk at hand, we can easily distinguish the vertices u and v. We consider L as an infinite sequence formed by an infinite concatenation of L. Let W_u be the closed walk in L of length $|L|$ starting from the first occurrence of u in L, and let W_v be the closed walk in L of length $|L|$ starting from the first occurrence of v in L. Obviously, because L has a period $|L|$, $W_u \neq W_v$, and thus these paths are distinguishing paths for u and v.

We now show that a Hamiltonian walk L of period $|L|$ exists in G_{vis}. This then implies that any two vertices of different holes are distinguishable.

Lemma 4. *The visibility graph G_{vis} of any polygon $\mathcal{P}$ with holes contains a Hamiltonian walk L of period $|L|$.*

Proof. We construct one such Hamiltonian walk as follows. Let v_1 be a vertex of $\bar{\mathcal{P}}$. We initially set L to be the walk from v_1 along the boundary of $\bar{\mathcal{P}}$. If $v_1, \ldots, v_{n_{\bar{\mathcal{P}}}}$ denote the vertices of the boundary of $\bar{\mathcal{P}}$, then initially $L = (v_1, \ldots, v_{n_{\bar{\mathcal{P}}}})$. Obviously, L is not Hamiltonian. We extend L as follows. We mark all vertices of $\bar{\mathcal{P}}$ as *visited*; all other vertices are marked as *unvisited*. For every vertex v_i of $\bar{\mathcal{P}}$ we compute, in the order as the vertices appear on the boundary (starting from v_1), a depth-first search tree in the graph induced by the unvisited vertices of G_{vis}. The depth-first search from v_i induces a closed walk $L(v_i)$ on the computed depth-first search tree. We add this walk into the walk L in the place of v. We mark all vertices from the depth-first search tree as visited and proceed with the next vertex v on the boundary of $\bar{\mathcal{P}}$.

We have computed a closed walk L in G_{vis} of the form $L(v_1), L(v_2), \ldots, L(v_{n_{\bar{\mathcal{P}}}})$. Obviously, the walk visits every vertex of G_{vis}.

Moreover, the walk has period $|L|$: Recall that we can identify v_1 and v_2 (as they are from the boundary of $\bar{\mathcal{P}}$); Observe that the occurrences of v_i come consecutively in L without being "interrupted" by another vertex v_j, $i \neq j$; Thus, we can uniquely identify the last occurrence of v_1 in L as it comes before the first occurrence of v_2; Thus, any two vertices u and v are distinguishable by the different distances from u and v to the last occurrence of v_1, respectively. □

Lemma 5. *No two vertices from different holes appear in the same class of G^*.*

Now, Lemma 2, Lemma 3, and Lemma 5 imply the main result of the paper:

Theorem 2. *The minimum base graph G^* is equal to the edge-labeled visibility graph G_{vis}.*

Theorem 3. *The robot can compute the visibility graph G_{vis} of any polygon $\mathcal{P}$ with holes, and it can localize its position in G_{vis}.*

Proof. The robot can compute the minimum base graph G^* of G_{vis} and its position therein using the algorithm in [7]. Theorem 2 implies that the computed graph G^* is actually what we want – the visibility graph G_{vis}. □

3 Conclusions

We have studied the mapping and localization problem by a simple robot inside a polygon $\mathcal{P}$ with a boundary compass. We have presented a black-box solution approach to show that such a robot can always compute the visibility graph G_{vis} of $\mathcal{P}$ whenever it knows an upper bound on the number of vertices of $\mathcal{P}$. The central part of the black-box approach is to prove that the minimum base graph G^* of G_{vis} is the visibility graph G_{vis}, i.e., $G^* = G_{\text{vis}}$. Our algorithm uses the generic algorithm of Chalopin et al. [7] for computing the minimum base graph of any edge-labeled directed graph G by a robot. This algorithm has an exponential running time in the worst case. We leave it open whether the time complexity can be improved for a robot with boundary compass.

Due to the fundamental importance of the mapping problem, our solution has further implications for other tasks. For example, it follows that a collection of robots with boundary compass and knowledge of an upper bound on the number of vertices can solve the *strong rendezvous* problem, i.e., they can meet in a vertex of $\mathcal{P}$ (even in an asynchronous model with no communication between the robots).

References

1. Ando, H., Oasa, Y., Suzuki, I., Yamashita, M.: Distributed memoryless point convergence algorithm for mobile robots with limited visibility. IEEE Transactions on Robotics and Automation 15(5), 818–828 (1999)

2. Bender, M., Fernandez, A., Ron, D., Sahai, A., Vadhan, S.: The power of a pebble: Exploring and mapping directed graphs. In: Proceedings of the 30th ACM Symposium on Theory of Computing, pp. 269–287 (1998)
3. Bilò, D., Disser, Y., Mihalák, M., Suri, S., Vicari, E., Widmayer, P.: Reconstructing visibility graphs with simple robots. Theoretical Computer Science 444, 52–59 (2012)
4. Boldi, P., Vigna, S.: Fibrations of graphs. Discrete Mathematics 243(1-3), 21–66 (2002)
5. Brunner, J., Mihalák, M., Suri, S., Vicari, E., Widmayer, P.: Simple robots in polygonal environments: A hierarchy. In: Proceedings of the Fourth International Workshop on Algorithmic Aspects of Wireless Sensor Networks, pp. 111–124 (2008)
6. Chalopin, J., Das, S., Disser, Y., Mihalák, M., Widmayer, P.: Mapping simple polygons: How robots benefit from looking back. Algorithmica (2011)
7. Chalopin, J., Das, S., Disser, Y., Mihalák, M., Widmayer, P.: Telling convex from reflex allows to map a polygon. In: Proceedings of the 28th International Symposium on Theoretical Aspects of Computer Science, pp. 153–164 (2011)
8. Chen, D., Wang, H.: An improved algorithm for reconstructing a simple polygon from the visibility angles. Computational Geometry: Theory and Applications 45, 254–257 (2012)
9. Disser, Y.: Mapping Polygons. Ph.D. thesis, ETH Zurich (2011)
10. Disser, Y., Mihalák, M., Widmayer, P.: Reconstruction of a polygon from angles without prior knowledge of the size. Tech. Rep. 700, ETH Zürich, Institute of Theoretical Computer Science (November 2010)
11. Disser, Y., Mihalák, M., Widmayer, P.: A polygon is determined by its angles. Computational Geometry: Theory and Applications 44, 418–426 (2011)
12. Donald, B.R.: On information invariants in robotics. Artificial Intelligence 72(1-2), 217–304 (1995)
13. Ganguli, A., Cortés, J., Bullo, F.: Distributed deployment of asynchronous guards in art galleries. In: Proceedings of the 2006 American Control Conference, pp. 1416–1421 (2006)
14. Gfeller, B., Mihalak, M., Suri, S., Vicari, E., Widmayer, P.: Counting targets with mobile sensors in an unknown environment. In: Proceedings of the 3rd International Workshop on Algorithmic Aspects of Wireless Sensor Networks, pp. 32–45 (2007)
15. Ghosh, S.K.: Visibility Algorithms in the Plane. Cambridge University Press (2007)
16. Ghosh, S.K., Goswami, P.P.: Unsolved problems in visibility graph theory. In: Proceedings of the India-Taiwan Conference on Discrete Mathematics, pp. 44–54 (2009)
17. Katsev, M., Yershova, A., Tovar, B., Ghrist, R., LaValle, S.M.: Mapping and pursuit-evasion strategies for a simple wall-following robot. IEEE Transactions on Robotics 27(1), 113–128 (2011)
18. O'Kane, J.M., LaValle, S.M.: On comparing the power of robots. International Journal of Robotics Research 27(1), 5–23 (2008)
19. Suri, S., Vicari, E., Widmayer, P.: Simple robots with minimal sensing: From local visibility to global geometry. International Journal of Robotics Research 27(9), 1055–1067 (2008)

Metric Dimension for Gabriel Unit Disk Graphs Is NP-Complete

Stefan Hoffmann and Egon Wanke*

Institute of Computer Science, Heinrich-Heine-Universität Düsseldorf, Germany
stefan.hoffmann@uni-duesseldorf.de,
wanke-algosensors@acs.uni-duesseldorf.de

Abstract. We show that finding a minimal number of landmark nodes for a unique virtual addressing by hop-distances in wireless ad-hoc sensor networks is NP-complete even if the networks are unit disk graphs that contain only Gabriel edges. This problem is equivalent to Metric Dimension for Gabriel unit disk graphs.

Wireless radio networks in which all nodes have the same radio range are often modeled as *unit disc graphs*. The vertices represent the sensor nodes and the undirected edges the symmetric communication channels between them. A widely used idea to reduce the complexity of distributed algorithms for sensor networks is to consider only some of the available connections. The strongest restriction that preserves connectivity is a spanning tree that unfortunately does not allow efficient routing through the network. A slightly weaker restriction is the *Gabriel graph*.

Many routing algorithms for sensor networks, for example BVR [3], are based on *virtual coordinates* computed from the distances to specific *landmark nodes* that flood large parts of the network. Afterwards every node defines its virtual coordinates depending on the distances to them. To minimize initialization overhead and address length the number of landmarks should be as small as possible, but on the other hand all virtual addresses should be unique amongst the network.

A set of vertices $S \subseteq V$ of a graph $G = (V, E)$ is called a *resolving set*, if for every vertex pair $u, v \in V$ there is at least one vertex $s \in S$ such that the length of shortest path, also called *hop-distance* $d(s, u)$, between s and u differs from the hop-distance $d(s, v)$ between s and v.

For a given graph G and an integer k the question of whether there is a resolving set with at most k vertices is called Metric Dimension and this problem is known to be NP-complete for general graphs as well as for planar graphs [2]. It is decidable in polynomial time for special classes of graphs like trees, wheels, complete graphs and k-regular bipartite graphs.

An undirected graph $G = (V, E)$ is called a *unit disc graph (UDG)*, if its vertices can be embedded into the euclidean plane such that there is an edge $\{u, v\} \in E$ if and only if the euclidean distance between the positions of u and v is at most 1. It is called a *Gabriel UDG (GUDG)* if there is such an embedding in that all edges additionally are *Gabriel edges*. An edge $\{u, v\} \in E$ is a Gabriel

A. Bar-Noy and M.M. Halldórsson (Eds.): ALGOSENSORS 2012, LNCS 7718, pp. 90–92, 2013.

edge if the smallest circle that contains the positions of u and v does not contain any other vertex positions. The restriction to Gabriel edges is a well known and reasonable type of topology control. They induce a planar subnetwork of G which is an $O(\sqrt{n})$ distance and optimal energy spanner [1].

Theorem 1. GUDG METRIC DIMENSION *is NP-complete.*

The NP-hardness is shown by a reduction from 1-NEGATIVE PLANAR 3-SAT [2]. The proof follows the same basic idea as the proof for planar graphs in [2], using similar graphs, called *gadgets*, to construct the graph H for an instance of GUDG METRIC DIMENSION. However, this proof is not a straightforward refinement of the proof for planar graphs, it requires more different gadgets to cover all possibilities as well as a gadget for edges of the clause variable graph. The *clause variable graph* G_ψ for an instance $\psi = (X, \mathcal{C})$ of 1-NEGATIVE PLANAR 3-SAT contains one vertex for every variable $x \in X$ and every clause $c \in C$ as well as an edge (x, c), if and only if c contains a literal of variable x. It is one of the restrictions in 1-NEGATIVE PLANAR 3-SAT that G_ψ is a planar graph.

There also is a preprocessing phase involved that modifies ψ and a *planar, orthogonal grid drawing (POGD)* of its clause variable graph G_ψ into an equivalent instance ψ' and a POGD of its clause variable graph $G_{\psi'}$ that has an additional property needed for the construction. A POGD of a directed, planar graph (V, E) is an embedding of its vertices onto integral positions together with a set of edge paths, one for every edge in E. An *edge path* is a sequence of points that defines how the edges are drawn by straight, unit length edge segments, each of which parallel to either the x- of the y-axis of the coordinate system, in a way that different edges to not intersect each other except at a common end point. Every directed, planar graph with vertex degree at most four has a POGD that can be computed in linear time [4]. The preprocessing phase alters ψ and an arbitrary PODG of G_ψ in order to get an equivalent instance ψ' and a POGD of $G_{\psi'}$ that does not contain any edge paths consisting of more than two edge segments.

The GUDG H is assembled by adding a copy of one of the gadgets for every variable vertex, clause vertex and edge of $G_{\psi'}$. These copies are then connected by identifying special t, f-vertex pairs with each other.

To ensure that H is a GUDG we present multiple GUDG embeddings, called tiles, for all gadgets into standardized polygons that can simply be connected with each other due to fixed positions for the t, f-vertex pairs. To compute a GUDG embedding for H the tiles are placed onto the positions of the vertices and edge paths in the POGD of $G_{\psi'}$.

However, to obtain a valid GUDG embedding, the correct tile has to be chosen for every copy of a gadget: Every tile maps all t, f-vertex pairs of its gadget to specific positions, but due to the identification of two t, f-vertex pairs with each other during the construction of H every one of these vertex pairs belongs to two different gadgets and therefore might be mapped to two different positions, resulting in an invalid embedding. Furthermore, since there are two different gadgets for variables that occur as a literal in exactly two clauses that have

different tiles, the assembly of H depends on this tile selection, that therefore has to be computed in polynomial time during the reduction.

To conclude the proof of Theorem 1 we then show that ψ is satisfiable if and only if there is a resolving set for H of size at most $4 \cdot |X'|$, where $|X'|$ is the number of variables in ψ'. This is accomplished by first proving that a specific set of vertices, so called *forced landmarks*, has to be a subset of every resolving set for H. Afterwards it is proven that these forced landmarks provide almost unique addressing for the network, except for one pair of vertices inside each variable and clause gadget copy that have the same hop-distance to all forced landmarks. This *address conflict* in each gadget results in the necessity to select one more landmark in every variable gadget copy, because the address conflict inside a variable gadget copy can only be resolved by choosing another landmark inside the same copy. Depending on which vertex is chosen the corresponding variable is set either to true or to false and it will also resolve the address conflict in adjacent clause gadget copies accordingly.

References

1. Bose, P., Devroye, L., Evans, W.S., Kirkpatrick, D.G.: On the spanning ratio of gabriel graphs and β-skeletons. SIAM Journal on Discrete Mathematics 20(2), 412–427 (2006)
2. Díaz, J., Pottonen, O., Serna, M., van Leeuwen, E.J.: On the Complexity of Metric Dimension. In: Epstein, L., Ferragina, P. (eds.) ESA 2012. LNCS, vol. 7501, pp. 419–430. Springer, Heidelberg (2012)
3. Fonseca, R., Ratnasamy, S., Zhao, J., Ee, C.T., Culler, D.E., Shenker, S., Stoica, I.: Beacon vector routing: Scalable point-to-point routing in wireless sensornets. In: NSDI. USENIX (2005)
4. Tamassia, R., Tollis, I.G.: Planar grid embedding in linear time. IEEE Transactions on Circuits and Systems 36(9), 1230–1234 (1989)

Pseudo-scheduling: A New Approach to the Broadcast Scheduling Problem

Shaun N. Joseph[1] and Lisa C. DiPippo[2]

[1] mZeal Communications*, Littleton MA 01460 USA
shaun.joseph@mzeal.com
[2] University of Rhode Island, Kingston RI 02881 USA
dipippo@cs.uri.edu

Abstract. The broadcast scheduling problem asks how a multihop network of broadcast transceivers operating on a shared medium may share the medium in such a way that communication over the entire network is possible. This can be naturally modeled as a graph coloring problem via distance-2 coloring ($L(1,1)$-labeling, strict scheduling). This coloring is difficult to compute and may require a number of colors quadratic in the graph degree. This paper introduces *pseudo-scheduling*, a relaxation of distance-2 coloring. Centralized and decentralized algorithms that compute pseudo-schedules with colors linear in the graph degree are given and proved.

Keywords: broadcast scheduling, TDMA, FDMA, graph coloring, wireless networks, ad-hoc networks.

1 Introduction

The broadcast scheduling problem asks how an arbitrary multihop network of broadcast transceivers operating on a shared medium may share the medium in such a way that communication over the entire network is possible. In particular, two or more transmissions made simultaneously (in time and space) on the same medium should be expected to fail; ie, the transmissions conflict.

A medium access control (MAC) protocol is a practical solution to the broadcast scheduling problem. The predominant approach to MAC protocol design is contention, the outstanding example of which is carrier sense multiple access (CSMA). Examples include the wireless Ethernet standard 802.11 and the protocol B-MAC [6] for wireless sensor networks.

The alternative to contention is explicit scheduling, such as time-division multiple access (TDMA) or frequency-division multiple access (FDMA). Regardless of how the medium is divided, however, the allocation of quanta to network nodes is naturally expressed as graph coloring problem; eg, a graph coloring with ten colors might correspond to a TDMA frame with ten timeslots.

* Research done while at the University of Rhode Island.

A. Bar-Noy and M.M. Halldórsson (Eds.): ALGOSENSORS 2012, LNCS 7718, pp. 93–104, 2013.

There are a variety of graph coloring problems extant, but the obvious and canonical model is the $L(1,1)$-*labeling*, also known as *distance-2 coloring*, *coloring of the graph square*, or *strict scheduling*. Here a vertex must be colored differently from any other vertex at distance one or two. The seminal results on this coloring were obtained by McCormick [5], who found that strict scheduling is NP-Complete as a decision problem; and that the number of colors required is $\Delta^2 + 1$ in the worst case, where Δ is the graph degree.

This paper introduces *pseudo-scheduling*, a relaxation of strict scheduling. Whereas strict schedules guarantee that every path in the graph is a valid communication path, pseudo-schedules only require the existence of some communication path between any two vertices; the requisite paths may exist along the edges of a spanning tree, in exact analogy to a network routing tree.

Pseudo-scheduling is defined formally as a graph coloring problem below. In §2 we survey related work. A centralized pseudo-scheduling algorithm using colors at most twice the graph degree is presented in §3, and in §4 we examine an algorithm that is decentralized but still uses colors only linear in the graph degree (with a reasonable multiplicative factor). We conclude in §5.

1.1 Definitions

Let us consider (vertex) coloring from the perspective of how colored vertices do or do not "conflict." Let $G = (V, E)$ be a simple, undirected graph with a coloring $l : V \to \mathbb{Z}$. We say that the ordered pair $(u, v) \in V^2$ is *nonconflicting* iff $uv \in E$, $l(u) \neq l(v)$, and for all $x \neq u$ adjacent to v, $l(u) \neq l(x)$. A directed path from u to v is likewise nonconflicting iff the pairs that comprise it are nonconflicting.

Observe that a strict schedule can be defined as a coloring such that every path in the graph is nonconflicting. Immediately we conceive of a natural relaxation: instead of requiring that every path be nonconflicting, demand only the existence of at least one nonconflicting path from u to v for every u, v that are connected in G. Such a coloring we call a *pseudo-schedule*.

It is convenient to work with edges uv such that both (u, v) and (v, u) are nonconflicting (under some coloring); such an edge is said to be *bidirectional*. A subgraph is bidirectional iff its edges are bidirectional. A pseudo-schedule with a bidirectional subgraph H is an *H-pseudo-schedule*.

A (symmetric link) network with a routing tree can be represented by a graph G with spanning tree T. A T-pseudo-schedule s of G is then a very interesting structure, as it ensures nonconflict along the routing tree but allows it elsewhere.

For the remainder of this paper, we assume that all graphs are simple; they may also be taken as connected without loss of generality. $V(G)$ and $E(G)$ denote the vertex and edge sets of G, respectively. Δ_G denotes the degree of graph G, $\deg_G(v)$ the degree of a vertex in G, and $N_G[v]$ the closed neighborhood, that is, the set comprised of v and all vertices adjacent to v in G. Finally, $\text{dist}_G(u, v)$ is the graph distance between vertices u, v in G.

2 Related Work

Although contention remains the most common approach to solving the broadcast scheduling problem, a number of explicit scheduling algorithms and protocols have been proposed. These can be separated into two categories: node-oriented scheduling, which is naturally modeled by vertex coloring, and link-oriented scheduling, modeled by edge coloring. We consider only the former, in particular because we wish to exploit one-to-many broadcast transmissions.

Node-oriented scheduling has been held back by its approach to conflict avoidance, which has hitherto taken strict scheduling as its starting point. This is explicit in DRAND [9], which implements the greedy algorithm for strict scheduling. Alternately, conflicts are tolerated in Z-MAC [8] and Funneling-MAC [1], but only inasmuch as these protocols combine TDMA with CSMA; the TDMA part of the protocol is strict. TSMA [2] and RIMAC [3] also permit conflicts, but this is aimed primarily at making the schedule easier to compute via decentralized, probabilistic methods, rather than reducing the division of the medium; indeed, the division tends to increase.

In his work on RAC-CT [7], Ren exhibits and implements what is basically the greedy algorithm for pseudo-schedules. Although Ren gives empirical evidence that the algorithm uses a number of colors very close to the graph degree on random grid graphs, the general upper bound is quadratic in the graph degree, giving no asymptotic improvement over strict scheduling.

3 Twice-Degree Algorithm

If conditions permit the use of a centralized algorithm, and we are more or less indifferent to the choice of spanning tree, the algorithm presented in this section will produce a pseudo-schedule of any graph G in no more than $2\Delta_G$ colors. The user chooses a root vertex r, most likely corresponding to a base station/access point; the algorithm selects the spanning (routing) tree to its convenience, although this tree will minimize distances to r.

For any tree T rooted at r, we say that u is the *parent* of v and v is a *child* of u iff $uv \in E(T)$ and $\mathrm{dist}_T(u,r) < \mathrm{dist}_T(v,r)$; we write $u = \mathrm{par}_{T,r}(v)$. (It will be convenient to let $\mathrm{par}_{T,r}(r) = r$.) We also henceforth permit ourselves the following abuse of notation: given a coloring s, which may be only partially defined, for any set U of vertices let $s(U)$ denote the set $\{s(u); u \in U, s(u) \text{ is defined}\}$.

3.1 Analysis

It is fairly obvious that the algorithm will use at most $2\Delta_G$ colors; the only difficulty is to show that it actually produces a pseudo-schedule.

Theorem 1. *Let s be the coloring produced by the twice-degree algorithm with input (G,r); then s is a T-pseudo-schedule, where T is the tree generated internally by the algorithm.*

Input: G, a graph; and r, a distinguished vertex of G.
Output: A pseudo-schedule on G.

```
1  V_T ← {r}, E_T ← ∅
2  T ← (V_T, E_T)
3  Q ← {r}, Q' ← ∅ // Q, Q' are queues
4  s ← ∅
5  repeat
6      foreach v ∈ Q (FIFO) do
7          N ← N_G[v] − V_T
8          append N to Q' (in any order)
9          V_T ← V_T ∪ N
10         E_T ← E_T ∪ {vx; x ∈ N}
11         K ← s(N_G[par_{T,r}(v)])
12         foreach x ∈ N_G[v] do
13             if x ≠ v and vx ∉ E_T then
14                 add s(par_{T,r}(x)) to K
15         k ← min(ℤ⁺ − K)
16         add v ↦ k to s
17     Q ← Q', Q' ← ∅
18 until Q = ∅
19 return s
```

Algorithm 3.1: The twice-degree algorithm

Proof. Observe that T is produced by a breadth-first search process and that every vertex at r-distance i is colored before any vertex at r-distance $i+1$. We can also see that if vertices u, v have distinct parents p_u, p_v respectively, then if p_u was colored before p_v, u was colored before v.

Take $uv \in E(T)$ such that $u = \text{par}_{T,r}(v)$. First we show that (u, v) is nonconflicting. $s(u) \neq s(v)$ clearly. Consider next any child x of v; since u is adjacent to $v = \text{par}_{T,r}(x)$, we have $s(u) \neq s(x)$. The only vertices left to check are those in $N_G[v] - N_T[v]$; let y be such a vertex. Now if $\text{dist}_G(r, u) < \text{dist}_G(r, y)$, then y was colored after u, so $s(u) \neq s(y)$. If, on the other hand, $\text{dist}_G(r, u) = \text{dist}_G(r, y)$, then u "adopted" v before y could (line 10), which implies that u was colored before y, hence $s(y) \neq s(x)$.

Let us now establish that (v, u) is nonconflicting. Obviously $s(v) \neq s(x)$ for any $x \in N_G[u]$ with $\text{dist}_G(r, x) < \text{dist}_G(r, v)$ since in this case x must have been colored before v. Turning to $x \in N_G[u]$ with $\text{dist}_G(r, x) = \text{dist}_G(r, v)$, clearly $s(v) \neq s(x)$ if x is a child of u, since v will be checked before coloring x and vice-versa. If, on the other hand, $\text{par}_{T,r}(x) = w \neq u$, it must be that w was colored before u since the former adopted x, thus $s(x)$ was defined when the algorithm computed $s(N_G[u])$ (line 11) before coloring v, and $s(v) \neq s(x)$. □

4 d-Band Algorithm

The twice-degree algorithm employs a very small number of colors in the worst case, but the requirements for central control and user indifference to the resulting

spanning tree may not be reasonable in network applications. The d-band algorithm does away with these requirements, albeit at the cost of raising the worst-case number of colors used, although this remains linear in the graph degree (with a reasonable coefficient).

Let G be a graph with a spanning tree T rooted at r. Intuitively, the algorithm divides the graph into d bands based on vertices' T-distance from r modulo d, with each band being colored from its own palette, disjoint from every other. The idea is that, if d is sufficiently large, we can rule out many conflicts *a priori*, greatly reducing the number of vertices that have to be checked.

The d-band algorithm is decentralized, with each vertex acting as an autonomous agent passing the following messages:

- REQ-COL(L), where L is a set of excluded colors;
- PUT-COL(x,k), where x is a vertex being assigned color k;
- RPT-COL(k,w), where k is the sender's color (if known) and w is a vertex that must be colored before the sender is colored; or, in a "reverse report," where k is a color excluded for the sender;
- RPT-PAR(k,w), where k is the color of the sender's parent (if known) and w is a vertex that must be colored before the sender's parent is colored; or, in a "reverse report," where k is the color of w, a stepparent of the sender;
- DEP-REQ(w), where w is a vertex whose color must be assigned before the sender can issue REQ-COL; and
- DEP-PUT(w), where w is a vertex whose color must be assigned before the sender can issue PUT-COL.

The sending vertex is implicitly included in any message, along with information about the intended receiver. We let ∞ denote an unknown color and let

$$\text{Palette}(v) = \{\text{dist}_T(r, v) \bmod d + id; i \geq 0\}. \tag{1}$$

Along with the definitions of parent and child as in §3, we say also that u is a *stepparent* of v iff $\text{dist}_T(u, r) = \text{dist}_T(v, r) - 1$ and $uv \in E(G) - E(T)$; and that x is a *stepchild* of y iff $\text{dist}_T(x, r) = \text{dist}_T(y, r) + 1$ and $xy \in E(G) - E(T)$. Each vertex is assumed to know its parent, children, stepparents, and stepchildren. Additionally, each vertex knows its T-distance from r. Finally, we assume that $V(G)$ admits a strict total order $\prec$ that can be efficiently computed at any vertex.

The flow of the algorithm about a vertex v can be sketched roughly as follows:

1. v listens for RPT-PARs from all of its stepchildren, building a list of excluded colors L.
2. v sends REQ-COL(L) to its parent u.
3. u listens for RPT-COLs from all of its stepchildren, building a list of forbidden colors K.
4. u sends PUT-COL(v,k_v) to v (and all stepchildren of u), where k_v is the smallest color in the palette of v not in $K \cup L$.

5. v broadcasts RPT-COL(k_v,v).
6. Each child of v sends RPT-PAR(k_v,v) to all its stepparents.

(In this sketch, for the sake of simplicity we have ignored the DEP facility.) In general, the d-band algorithm colors the leaves of T first and proceeds towards the root, although significant parallelism is possible.

The root vertex r assigns itself the color 0, making this known by sending RPT-COL(0,r) to its children. Any vertex besides r must acquire its color as per AcquireColor (Algorithm 4.1). Any vertex with children must assign colors to its children as per AssignColors (Algorithm 4.2). Finally, any non-root vertex must receive and relay reports as per ReportColors (Algorithm 4.3). The ensemble of these procedures, running independently and in parallel on every vertex simultaneously, constitutes the d-band algorithm.

(We assume fully reliable transmission with synchronous communication primitives *send* and *listen*. AssignColors uses the primitive *ack-send* $\mathcal{M}$ *to* x by which is meant: send $\mathcal{M}$ to x and wait until $\mathcal{M}$ is sent back as confirmation, queuing any messages that arrive in the meanwhile for retrieval by the next call to listen.)

4.1 Analysis

We say that the d-band algorithm *terminates* on a particular graph with rooted spanning tree iff AcquireColor returns on every vertex.

Theorem 2. *The d-band algorithm terminates on any graph with any choice of root and spanning tree.*

Proof. Let G be a graph with spanning tree T rooted at r. AcquireColor (Algorithm 4.1) on vertex v does its main work in the loop beginning at line 6. As this loop is bypassed when v has no stepchildren, assume that it does. We say that v has a *request dependence* on u when u is the parent of a stepchild of v; and just as v depends on u, u may depend on t, and so on. If we can follow the dependency chain to some terminal a that has no stepchildren, there is no problem, since we can inductively work back to v. However, the dependency chain may in fact be a cycle, in the sense that v has request dependence on u, u has request dependence on t, and so on up to a, but then a has request dependence on v. This we call a *dependency cycle of type I.*

Given C, a dependency cycle of type I, let $v = \min_{\prec} C$. Assume, for the time being, that C is the only dependency cycle in the graph. v issues DEP-REQ(v) to its children, and via ReportColors (Algorithm 4.3) one of the children sends RPT-PAR(∞,v) to x, which depends on v. But since $v \prec x$, x issues DEP-REQ(v) to its children, one of which then sends RPT-PAR(∞,v) to y, which depends on x, and so on.

Let u be the vertex in C on which v depends, creating the cycle. u issues DEP-REQ(v) to its children, and one of them transmits RPT-PAR(∞,v) to v. At this point v can detect the dependency cycle, and v breaks the cycle by ignoring its dependence on u (see line 12 of AcquireColor). As per our assumptions, v is now free of request dependencies, or at worst sits in linear dependence chains that

are naturally resolved; that is, v (eventually) acts as if it has no stepchildren, and proceeds to issue REQ-COL to its parent p.

Let us assume that p eventually assigns a color to v via PUT-COL. v then broadcasts RPT-COL, resolving the now-linear dependency chain. (The resolution is a little unusual at u, where we have registered a "reverse dependence" on v—but this will be cleared by the RPT-COL broadcast from v, which causes a "reverse report" RPT-PAR to be sent to u from one of its children.) Hence every vertex in C gradually becomes free to issue REQ-COL, and if we assume that every one of their parents replies with PUT-COL, then AcquireColor terminates on every vertex in C.

We now shift our attention to AssignColors (Algorithm 4.2). A vertex v with parent p_v is said to have a *put dependence* on any stepchild of p_v. (It is convenient for the dependence to be registered at p_v.) Just like request dependencies, put dependencies can be chained and may form a cycle; this we call a *dependency cycle of type II*. It is not hard to see that a type II cycle is broken by essentially

```
   Input: v, "this" vertex.
   Output: A color.
 1 L ← ∅
 2 f_W ← {x ↦ v; x is a stepchild of v}
 3 W ← Image(f_W)
 4 w_≺ ← v
 5 send RPT-COL(∞,v) to children of v
 6 while W ≠ ∅ do
 7     listen for message M
 8     if M = RPT-PAR(k,w) from (step)child x of v then
 9         if k ≠ ∞ then
10             remove x ↦ f_W(x) from f_W
11             add k to L
12         else if w = v then remove x ↦ f_W(x) from f_W
13         else add/replace x ↦ w in f_W
14         W ← Image(f_W)
15     else if M = DEP-REQ(w) from a child of v then
           // Reverse dependence
16         add/replace w ↦ w in f_W
17     if w_≺ ≠ min_≺(W ∪ {v}) then
18         w_≺ ← min_≺(W ∪ {v})
19         send DEP-REQ(w_≺) to children of v
20         send RPT-COL(∞,w_≺) to stepparents of v
21 send RPT-COL(∞,v) to children of v
22 send REQ-COL(L) to the parent of v
23 listen for PUT-COL(v,k_v) from the parent of v
24 send RPT-COL(k_v,v) to children, stepchildren, and stepparents of v
25 return k_v
```

Algorithm 4.1: The d-band algorithm: AcquireColor

```
   Input: v, "this" vertex.
 1 K, f_L, f_W, W ← ∅
 2 X ← stepchildren of v, Z ← children of v
 3 f_R ← {z ↦ ∅; z ∈ Z}
 4 w_≺ ← v
 5 while Z ≠ ∅ do
 6     if X = ∅ then
 7         foreach z ↦ L ∈ f_L such that f_R(z) = ∅ do
 8             k_z ← min(Palette(z) − K − L)
 9             send PUT-COL(z,k_z) to z and stepchildren of v
10             remove z from Z
11             add k_z to K
12     else if W ≠ ∅ and w_≺ ≠ min_≺ W then
13         w_≺ ← min_≺ W
14         send DEP-PUT(w_≺) to children of v
15     listen for message M
16     if M = REQ-COL(L) from child z of v then
17         if f_L(z) defined then L ← L ∪ f_L(z)
18         add/replace z ↦ L in f_L
19     else if M = RPT-COL(k,w) from (step)child x then
20         if x ∈ X then
21             if k ≠ ∞ or w is a child of v then
22                 remove x from X
23                 remove x ↦ f_W(x) from f_W
24                 add k to K
25                 if w is a child of v then ack-send DEP-PUT(v) to x
26             else add/replace x ↦ w to f_W
27             W ← Image(f_W)
28         else if x is a child of v then
29             if k ≠ ∞ then
30                 L ← (f_L(x) defined ? f_L(x) : ∅)
31                 add k to L
32                 add/replace x ↦ L in f_L
33             else remove w from f_R(z)
34     else if M = DEP-PUT(w) from child z of v then
           // Reverse dependence
35         add w to f_R(z)
36 send PUT-COL(∞,v) to stepchildren of v
```

Algorithm 4.2: The d-band algorithm: AssignColors

```
   Input: v, "this" vertex.
 1 p ← parent of v
 2 W_I, W_II ← ∅
 3 listen for message M from (step)parents
 4 if M = DEP-REQ(w) from p then
 5     if w is a stepparent of v then
           // Type I cycle breaking: reverse dependence
 6         add w to W_I
 7         send DEP-REQ(w) back to p
 8     send RPT-PAR(∞,w) to stepparents of v
 9 else if M = RPT-COL(k,w) from x then
10     if x = p then
11         send RPT-PAR(k,w) to stepparents of v
12     else if x ∈ W_I and k ≠ ∞ then
           // Type I cycle breaking: reverse report
13         remove x from W_I
14         send RPT-PAR(k,x) to p
15 else if M = DEP-PUT(w) from x then
16     if x = p then
17         send RPT-COL(∞,w) to children and stepparents of v
18     else
           // Type II cycle breaking: reverse dependence
19         add x to W_II
20         send DEP-PUT(x) to p and x
21 else if M = PUT-COL(u,k) from x ∈ W_II then
       // Type II cycle breaking: reverse report
22     if k = ∞ then remove x from W_II
23     send RPT-COL(k,x) to p
```

Algorithm 4.3: The d-band algorithm: ReportColors

the same method used for the type I cycle, with DEP-PUT and RPT-COL standing in for DEP-REQ and RPT-PAR, respectively. (Once again, there is a special "reverse dependence" facility. Let p_v have stepchild u with parent p_u. If a type II cycle is broken at p_v, then p_u will register the reverse dependence of u on the children of p_v. Resolution comes when p_u finishes coloring its children, with "reverse report" RPT-COLs being sent to p_u via u.) Hence the d-band algorithm terminates in the presence of a dependency cycle of type II, provided that REQ-COL is issued.

Finally, a dependency cycle of *mixed type* is possible. AcquireColor handles the transition from request to put dependency by repackaging RPT-PAR as RPT-COL (line 20), while ReportColors handles the reverse transition by first repackaging DEP-PUT as RPT-COL (line 17) and then relaying the latter as RPT-PAR (line 11). But observe that a cycle of mixed type can be broken by the methods previously described; it is treated exactly as if it were a cycle of type I or type II if it is broken by AcquireColor or AssignColors, respectively.

The same mechanisms then assure that the resolution proceeds across the cycle. We conclude that the d-band algorithm terminates if there is no more than one dependency cycle in the graph (of whatever type).

Given a dependency cycle C (of any type), observe that there exists l such that $\text{dist}_T(v, r) = l$ for all $v \in C$; call l the level of C. Clearly cycles with different levels cannot affect each other; additionally, disjoint cycles do not interact. Thus the d-band algorithm terminates given any number of disjoint dependency cycles per level of T.

Unfortunately a graph may contain many overlapping dependency cycles; we claim the algorithm terminates regardless. Let $\mathcal{C}$ be a family of intersecting dependency cycles. As there is a strict total order $\prec$ on vertices, there exists some

$$v = \min_{\prec} \bigcup_{C \in \mathcal{C}} C.$$

Observe that AcquireColor must terminate on v, since all dependency cycles in $\mathcal{C}$ containing v will be broken at v, if not elsewhere. After breaking all such cycles and resolving all newly-linear chains, let $\mathcal{C}'$ be the remaining cycles. Obviously $|\mathcal{C}'| < |\mathcal{C}|$, and we can apply the same argument to $\mathcal{C}'$ inductively. We conclude that the d-band algorithm terminates. □

Because the d-band algorithm terminates, we can treat the color output by AcquireColor on every vertex as a coloring of the whole graph. However, d must be taken sufficiently large for coloring to be a pseudo-schedule, as made precise by the following theorem, the proof of which appears in §3.4.1 of [4]. Note that an immediate consequence of this is that $d = 3$ suffices if T is a shortest path tree.

Theorem 3. *Let G be a graph with spanning tree T rooted at r. The d-band algorithm yields a T-pseudo-schedule provided that*

$$d \geq \max_{uv \in E(G)} |\text{dist}_T(u, r) - \text{dist}_T(v, r)| + 2 \tag{2}$$

or d is greater than the height of T.

We will consider the number of colors used by a d-band pseudo-schedule s to be the greatest integer color appearing in s plus one; this forces us to account for "gaps" of unused colors.

Theorem 4. *Let h denote the number of colors used by the d-band algorithm on graph G with spanning tree T rooted at r, where d meets the conditions of Theorem 3. If $d \leq \text{height}(T) + 1$ and $\Delta_G \geq 2$, then*

$$d(\Delta_T - 1) + 1 \leq h \leq 2d(\Delta_G - 1) \tag{3}$$

except possibly when $\Delta_T = 2$, in which case the lower bound falls to d.

Proof. The lower bound for $\Delta_T = 2$ is established by any path graph. For $\Delta_T \geq 3$, let v be the vertex on which T achieves its maximum degree. Then any child of v uses a palette containing at least the colors $a, a+d, \ldots, a+d(\Delta_T - 1)$. With $a = 0$, we obtain the lower bound.

For the upper bound, consider a vertex v with $\text{dist}_T(v, r) = d - 1$. Its parent p_v has at most $\Delta_G - 2$ neighbors distinct from v but at the same T-level as v. Additionally, v has at most $\Delta_G - 1$ stepchildren, each of which could have a distinct parent. This is a total of $2\Delta_G - 2$ vertices (including v), so

$$\text{Palette}(v) \subseteq \{d-1, 2d-1, \ldots, d-1+d(2\Delta_G - 3)\}$$

which yields the upper bound. □

5 Conclusion

This work has introduced and motivated pseudo-scheduling as a new approach to the broadcast scheduling problem. The algorithms exhibited here prove that pseudo-scheduling can result in asymptotic improvements in medium division relative to strict scheduling. This would correspond concretely, for instance, to TDMA frames that grow only linearly with the neighborhood size, improving the network throughput, especially in ad-hoc wireless networks in which the neighborhood size cannot be tightly bounded in advance. Although the concepts have yet to be put to the test in practical network applications, a strong theoretical foundation now exists for implementers.

References

1. Ahn, G., Miluzzo, E., Campbell, A.T., Hong, S., Curomo, F.: Funneling-MAC: A localized, sink-oriented MAC for boosting fidelity in sensor networks. In: Fourth ACM Conference on Embedded Network Sensor Systems (SenSys 2006), pp. 293–306 (November 2006)
2. Chlamtac, I., Faragó, A., Zhang, H.: Time-spread multiple access protocols for multihop mobile radio networks. IEEE/ACM Transactions on Networking 5(6), 804–812 (1997)
3. DiPippo, L., Tucker, D., Fay-Wolfe, V., Bryan, K.L., Ren, T., Day, W., Murphy, M., Henry, T., Joseph, S.: Energy-efficient MAC for broadcast problems in wireless sensor networks. In: Third International Conference on Networked Sensing Systems (June 2006)
4. Joseph, S.N.: Relaxations of $L(1,1)$-labeling for the broadcast scheduling problem. Ph.D. thesis, University of Rhode Island (2011), http://digitalcommons.uri.edu/cgi/viewcontent.cgi?article=2382&context=dissertations
5. McCormick, S.T.: Optimal approximation of sparse Hessians and its equivalence to a graph coloring problem. Mathematical Programming 26, 153–171 (1983)
6. Polastre, J., Hill, J., Culler, D.: Versatile low power media access for wireless sensor networks. In: Second ACM Conference on Embedded Network Sensor Systems (SenSys 2004), pp. 95–107 (November 2004)

7. Ren, T.: Graph Coloring Algorithms for TDMA Scheduling in Wireless Sensor Networks. Ph.D. thesis, University of Rhode Island (2007)
8. Rhee, I., Warrier, A., Aia, M., Min, J.: Z-MAC: a hybrid MAC for wireless sensor networks. In: Third ACM Conference on Embedded Network Sensor Systems (SenSys 2005), pp. 90–101 (November 2005)
9. Rhee, I., Warrier, A., Min, J., Xu, L.: DRAND: distributed randomized TDMA scheduling for wireless ad-hoc networks. In: Seventh ACM International Symposium on Mobile Ad Hoc Networking and Computing (MobiHoc 2006), pp. 190–201 (May 2006)

Self-stabilizing TDMA Algorithms for Dynamic Wireless Ad-Hoc Networks*

Pierre Leone and Elad Michael Schiller

[1] Computer Science Department, University of Geneva, Geneva Switzerland
pierre.leone@unige.ch
[2] Chalmers University of Technology, Göteborg Sweden
elad@chalmers.se

We consider *Medium Access Control* (MAC) protocols for Dynamic wireless ad-hoc networks (DynWANs) that need to be autonomous, robust, and have high bandwidth utilization, a high predictability degree of bandwidth allocation, and low communication delay in the presence of frequent changes to the network topology. We propose an algorithmic design for self-stabilizing MAC protocols with a provable short convergence period, and by that, it can facilitate the satisfaction of severe timing requirements and possesses a greater predictability degree, while maintaining low communication delays and high throughput. We show that the algorithm facilitates the satisfaction of severe timing requirements for DynWANs. We consider transient faults and topological changes to the communication network, and demonstrate self-stabilization.

Algorithm Description. The MAC algorithm in Fig. 1 assigns timeslots to nodes after the convergence period. The system consists of a set, P, of N anonymous communicating entities, which we call *nodes*. Denote every node $p_i \in P$ with a unique index, i. We assume that the MAC protocol is invoked periodically by synchronized *common pulse* that aligns the starting time of the TDMA frame [1]. The term *(broadcasting) timeslot* refers to the period between two consecutive common pulses. In our pseudo-code, we use the event timeslot (t) that is triggered by the common pulse. Nodes raise the event carrier_sense() when they detect that the received energy levels have reached a threshold in which the radio unit is expected to succeed in carrier sense locking. We assume that timeslots allow the transmission of DATA packets using the transmit() and receive() primitives. Moreover, we consider *signaling (beacons)* as short packets that include no data load, rather their carrier sense delivers important information. Before the transmission of the DATA packet in timeslot t, the scheme uses beacons for singling the node intention to transmit the packet within t. During the convergence period several nodes can be assigned to the same timeslot. The algorithm solves such timeslot allocation conflicts by letting node p_i to go through a (listening/signaling) competition with and node p_j before transmitting in its broadcasting timeslot. The competition rules require each node to choose one out of n listening/signaling period for its broadcasting timeslot. This

* Detailed version of this paper is available as technical report [2]. This work was partially supported by the EC, through project FP7-STREP-288195, KARYON (Kernel-based ARchitecture for safetY-critical cONtrol).

A. Bar-Noy and M.M. Halldórsson (Eds.): ALGOSENSORS 2012, LNCS 7718, pp. 105–107, 2013.

```
   Constants, variables, macros and external functions
 2  MaxRnd (n in the proofs) : integer = bound on round number
    s : [0, T-1] ∪ {⊥} = next timeslot to broadcast or null, ⊥
 4  signal : boolean = trying to acquiring the channel
    unused[0, T-1] : boolean = marking unused timeslots
 6  unused_set = { k : unused[k] = true } : unused timeslot set (macro)
    MAC_fetch()/MAC_deliver() : MAC layer interface
 8  transmit/receive/carrier_sense : communication primitives

10 Upon timeslot(t)
    if t = 0 ∧ s = ⊥ then s := select_unused(unused_set)
12  (unused[t], signal) := (true, false) (* remove stale info. *)
    if s ≠ ⊥∧ t = s then send(MAC_fetch())
14
   Upon receive(< DATA, m> ) do MAC_deliver(< m> )
16
   Function send(m)   (* send message m to p_i's neighbors *)
18  for ((signal, k) := (true, 0); k := k + 1; k ≤ MaxRnd) do
      if signal then with probability ρ(k) = 1/(MaxRnd − k) do
20      signal := false (* quit the competition *)
        transmit(< BEACON> ) (* try acquiring the channel *)
22    wait until the end of competition round (* exposure period alignment *)
    if s ≠ ⊥ then transmit(< DATA, m> ) (* send the data packet *)
24
   Upon carrier_sense(t) (* defer transmission during t *)
26  if s = t ∧ signal then s := ⊥ (* mark that the timeslot is not unique *)
    (signal, unused[t]) := (false, false) (* quit the competition *)
28
   Function select_unused(set) (* select an empty timeslot *)
30  if set = ∅ then return ⊥ else return uniform_select(set)
```

Fig. 1. Self-stabilizing TDMA-based MAC algorithm, code of node p_i.

implies that among all the nodes that attempt to broadcast in the same timeslot, the ones that select the earliest listening/signaling period win this broadcasting timeslot and access the communication media. Before the winners access their timeslots, they signal to their neighbors that they won by sending beacons during their chosen signaling periods. When a node receives a beacon, it does not transmit during that timeslot, because it lost this competition. Instead, it randomly selects another broadcasting timeslot and competes for it on the next broadcasting round.

Discussion. Thus far, MAC algorithms could not consider timing requirements within a provably short recovery period that follows (arbitrary) transient faults and network topology changes. This work proposes the first self-stabilizing TDMA algorithm for DynWANs that has a provably short convergence period. Thus, the proposed algorithm possesses a greater degree of predictability, while maintaining low communication delays and high throughput.

The analysis shows that when there are N nodes in the network and $\alpha \in (0, 1)$, the network convergence time is bounded by equation (1) with probability $1-\alpha$, where n is the number of listening/signaling periods, d is the maximal node degree in the interference graph and T is the size of the TDMA frame. This means that with probability α all nodes are allocated with timeslots in maximum $k(n, N)$ broadcasting rounds, see Fig. 2.

$$\text{Convergence time:} \quad k(n,N) = 1 + \frac{log\left(1 - \sqrt[N]{1-\alpha}\right)}{log\left(1 - \left(\frac{n-1}{2n}\right)^{\frac{d}{T}}\right)} \tag{1}$$

Fig. 2 shows that the proposed algorithm demonstrates a low dependency degree on the number of nodes in the network even when considering $10,000$ nodes. We note that it uses merely a small fraction of the bandwidth that is spent on frame control information (say 3 listening/signaling periods) and when considering 99% probability to convergence within 30 to 35 TDMA frames.

The costs associated with predictable communications, say, using base-stations, motivate the adoption of new networking technologies, such as MANETs and VANETs. In the context of these technologies, we expect that our proposal would contribute to the development of MAC protocols that can be used by applications that need guarantees for severe timing requirements.

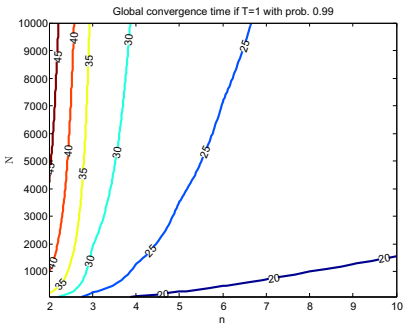


Fig. 2. Contour plot of equation (1) for $s = d/T = 1$. The contour lines connect values of $k(n, N)$ that are the same (see the text tags along the Contour lines). When N nodes attempt to access the medium, the conversance time is stable in the presence of a growing number, n, of listening/signaling periods.

References

1. Mustafa, M., Papatriantafilou, M., Schiller, E.M., Tohidi, A., Tsigas, P.: Autonomous TDMA alignment for VANETs. In: IEEE 76th Vehicular Technology Conference (VTC 2012-Fall), September 3-6 (2012)
2. Leone, P., Schiller, E.M.: Self-stabilizing TDMA algorithms for dynamic wireless ad-hoc networks. arXiv 1210.3061, October 12 (2012), http://arxiv.org/abs/1210.3061

Complexity of Connectivity in Cognitive Radio Networks through Spectrum Assignment*

Hongyu Liang[1], Tiancheng Lou[1], Haisheng Tan[1,**],
Amy Yuexuan Wang[1], and Dongxiao Yu[2]

[1] Institute for Theoretical Computer Science, Institute for Interdisciplinary Information Sciences, Tsinghua University, P.R. China
tan@tsinghua.edu.cn
[2] Department of Computer Science, The University of Hong Kong, Hong Kong, China

Abstract. Cognitive Radio Networks (CRNs) are considered as a promising solution to the spectrum shortage problem in wireless communication. In this paper, we address the algorithmic complexity of the connectivity problem in CRNs through spectrum assignment. We model the network of secondary users (SUs) as a potential graph, where if two nodes have an edge between them, they are connected as long as they choose a common available channel. In the general case, where the potential graph is arbitrary and SUs may have different number of antennae, we prove that it is NP-complete to determine whether the network is connectable even if there are only two channels. For the special case when the number of channels is constant and all the SUs have the same number of antennae, which is more than one but less than the number of channels, the problem is also NP-complete. For special cases that the potential graph is complete or a tree, we prove the problem is NP-complete and fixed-parameter tractable (FPT) when parameterized by the number of channels. Furthermore, exact algorithms are derived to determine the connectivity.

1 Introduction

Cognitive Radio is a promising technology to alleviate the spectrum shortage in wireless communication. It allows the unlicensed *secondary users* to utilize the temporarily unused licensed spectrums, referred to as *white spaces*, without interfering with the licensed *primary users*. Cognitive Radio Networks (CRNs) is considered as the next generation of communication networks and attracts numerous research from both academia and industry recently.

In CRNs, each secondary user (SU) can be equipped with one or multiple antennae for communication. With multiple antennae, a SU can communicate on multiple channels simultaneously (in this paper, channel and spectrum are

* This work was supported in part by the National Basic Research Program of China Grant 2011CBA00300, 2011CBA00302, the National Natural Science Foundation of China Grant 61073174, 61033001, 61061130540, and the Hi-Tech research and Development Program of China Grant 2006AA10Z216.

** Corresponding author.

A. Bar-Noy and M.M. Halldórsson (Eds.): ALGOSENSORS 2012, LNCS 7718, pp. 108–119, 2013.

used interchangeably.). Through spectrum sensing, each SU has the capacity to measure current available channels at its site, i.e. the channels are not used by the primary users (PUs). Due to the appearance of PUs, the available channels of SUs have the following characteristics [1]: 1) *Spatial Variation*: SUs at different positions may have different available channels; 2) *Spectrum Fragmentation*: the available channels of a SU may not be continuous; and 3) *Temporal Variation*: the available channels of a SU may change over time.

Spectrum assignment is to allocate available channels to SUs to improve system performance such as spectrum utilization, network throughput and fairness. Spectrum assignment is one of the most challenging problems in CRNs and has been extensively studied [12–15].

Connectivity is a fundamental problem in wireless communication. Connection between two nodes in CRNs is not only determined by their distance and their transmission powers, but also related to whether the two nodes has chosen a common channel. Due to the spectrum dynamics, communication in CRNs is more difficult than in the traditional multi-channel radio networks [3]. Authors in [8–10] studied the impact of different parameters on connectivity in large-scale CRNs, such as the number of channels, the activity of PUs, the number of neighbors of SUs and the transmission power.

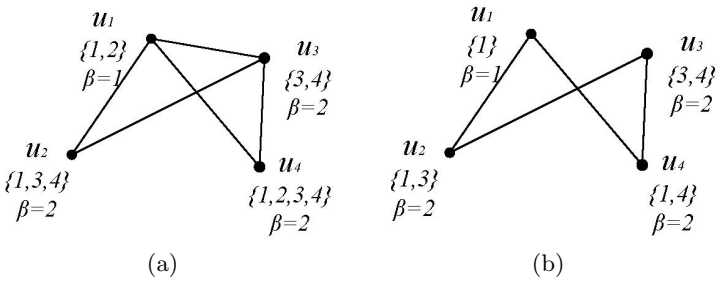


Fig. 1. the general case. a) the potential graph: the set besides each SU is its available channels, and β is its number of antennae. u_2 and u_4 are not connected because they are a pair of heterogenous nodes or their distance exceeds at least one of their transmission ranges. b) the realization graph which is connected: the set beside each SU is the channels assigned to it.

In this paper, we focus on the complexity of connectivity in CRNs through spectrum assignment. We model the network as a potential graph and a realized graph before and after spectrum assignment respectively (refer to Section 2). We start from the most general case, where the network is composed of heterogenous SUs[1], SUs may be equipped with different number of antennae and the potential graph can be arbitrary (Figure 1). Then, we proceed to study the special case when all the SUs have the same number of antennae. If all the SUs are

[1] We assume two heterogenous SUs cannot communicate even when they work on a common channel and their distance is within their transmission ranges.

homogenous with transmission ranges large enough, the potential graph will be a complete graph. For some hierarchically organized networks, e.g. a set of SUs are connected to an access point, the potential graph can be a tree. Therefore, we also study these special cases. Exact algorithms are also derived to determine connectivity for different cases. Our results are listed below. To the best of knowledge, this is the first work that systematically studies the algorithmic complexity of connectivity in CRNs with multiple antennae.

Our Contributions: In this paper we study the algorithmic complexity of the connectivity problem through spectrum assignment under different models. Our main results are as follows.

- When the potential graph is a general graph, we prove that the problem is NP-complete even if there are only two channels. This result is sharp as the problem is polynomial-time solvable when there is only one channel. We also design exact algorithms for the problem. For the special case when all SUs have the same number of antennae, we prove that the problem is NP-complete when $k > \beta \geq 2$, where k and β are the total amount of channels in the white spaces and the number of antennae on an SU respectively.
- When the potential graph is complete[2], the problem is shown to be NP-complete even if each node can open at most two channels. However, in contrast to the general case, the problem is shown to be polynomial-time solvable if the number of channels is fixed. In fact, we prove a stronger result saying that the problem is fixed parameter tractable when parameterized by the number of channels. (See [4] for notations in parameterized complexity.)
- When the potential graph is a tree, we prove that the problem is NP-complete even if the tree has depth one. Similar to the complete graph case, we show that the problem is fixed parameter tractable when parameterized by the number of channels.

Paper Organization: In Section 2 we formally define our model and problems studied in this paper. We study the problem with arbitrary potential graphs in Section 3. The special cases where the potential graph is complete or a tree are investigated in Sections 4 and 5. The paper is concluded in Section 6 with possible future works.

2 System Model and Problem Definition

We first describe the model used throughout this paper. A *cognitive radio network* is comprised of the following ingredients:

- U is a collection of secondary users (SUs) and C is the set of channels in the white spaces.
- Each SU $u \in U$ has a *spectrum map,* denoted by SpecMap(u), which is a subset of C representing the available channels that u can open.

[2] Note that the complete graph is a special case of disk graphs.

- The *potential graph* $\mathcal{PG} = (U, E)$, where each edge of E is also called a *potential edge*. If two nodes are connected by a potential edge, they can communicate as long as they choose a common available channel.
- Each SU $u \in U$ is equipped with a number of antennas, denoted as *antenna budget* $\beta(u)$, which is the maximum number of channels that u can open simultaneously.

For a set S, let 2^S denote the power set of S, i.e., the collection of all subsets of S. A *spectrum assignment* is a function $\mathcal{SA} : U \to 2^C$ satisfying that

$$\mathcal{SA}(u) \subseteq \textsc{SpecMap}(u) \text{ and } |\mathcal{SA}(u)| \leq \beta(u) \text{ for all } u \in U.$$

Equivalently, a spectrum assignment is a way of SUs opening channels such that each SU opens at most β channels and can only open those in its spectrum map.

Given a spectrum assignment $\mathcal{SA}$, a potential edge $\{u, v\} \in E$ is called *realized* if $\mathcal{SA}(u) \cap \mathcal{SA}(v) \neq \emptyset$, i.e., there exists a channel opened by both u and v. The *realization graph* under a spectrum assignment is a graph $\mathcal{RG} = (U, E')$, where E' is the set of realized edges in E. Note that $\mathcal{RG}$ is a spanning subgraph of the potential graph $\mathcal{PG}$. A cognitive radio network is called *connectable* if there exists a spectrum assignment under which the realization graph is connected, in which case we also say that the cognitive radio network is *connected* under this spectrum assignment. Now we can formalize the problems studied in this paper.

The Spectrum Connectivity Problem. The Spectrum Connectivity problem is to decide whether a given cognitive radio network is connectable.

We are also interested in the special case where the number of possible channels is small[3] and SUs have the same antenna budget. Therefore, we define the following subproblem of the Spectrum Connectivity problem:

The Spectrum (k, β)-Connectivity Problem. For two constants $k, \beta \geq 1$, the Spectrum (k, β)-Connectivity problem is to decide whether a given cognitive radio network with k channels in which all SUs have the same budget β is connectable. For convenience we write $\textsc{SpecCon}(k, \beta)$ to represent this problem.

Finally, we also consider the problem with special kinds of potential graphs, i.e. the potential graph is complete or a tree.

3 The Spectrum Connectivity Problem

In this section, we study the the Spectrum Connectivity problem from both complexity and algorithmic points of view.

3.1 NP-Completeness Results

We show that the Spectrum Connectivity problem is NP-complete even if the number of channels is fixed. In fact we give a complete characterization of the complexity of $\textsc{SpecCon}(k, \beta)$ by proving the following dichotomy result:

[3] Commonly, the white spaces include spectrums from channel 21 (512Mhz) to 51 (698Mhz) excluding channel 37, which is totally 29 channels [1].

Theorem 1. SpecCon(k, β) *is* NP*-complete for any integers* $k > \beta \geq 2$*, and is in* P *if* $\beta = 1$ *or* $k \leq \beta$*.*

The second part of the statement is easy: When $\beta = 1$, each SU can only open one channel, and thus all SUs should be connected through the same channel. Therefore, the network is connectable if and only if there there exists a channel that belongs to every SU's spectrum map (and of course the potential graph must be connected), which is easy to check. When $k \leq \beta$, each SU can open all channels in its spectrum map, and the problem degenerates to checking the connectivity of the potential graph.

In the sequel we prove the NP-completeness of SpecCon(k, β) when $k > \beta \geq 2$. First consider the case $k = \beta + 1$. We will reduce a special case of the Boolean Satisfiability (SAT) problem, which will be shown to be NP-complete, to SpecCon$(\beta + 1, \beta)$, thus showing the NP-completeness of the latter.

A clause is called *positive* if it only contains positive literals, and is called *negative* if it only contains negative literals. For example, $x_1 \vee x_3 \vee x_5$ is positive and $\overline{x_2} \vee \overline{x_4}$ is negative. A clause is called *uniform* if it is positive or negative. A *uniform* CNF formula is the conjunction of uniform clauses. Define Uniform-SAT as the problem of deciding whether a given uniform CNF formula is satisfiable.

Lemma 1. Uniform-SAT *is* NP*-complete.*

Proof. Let F be a CNF formula with variable set $\{x_1, x_2, \ldots, x_n\}$. For each i such that $\overline{x_i}$ appears in F, we create a new variable y_i, and do the following:

- substitute y_i for all occurrences of $\overline{x_i}$;
- add two clauses $x_i \vee y_i$ and $\overline{x_i} \vee \overline{y_i}$ to F. More formally, let $F \leftarrow F \wedge (x_i \vee y_i) \wedge (\overline{x_i} \vee \overline{y_i})$. This ensures $y_i = \overline{x_i}$ in any satisfying assignment of F.

Call the new formula F'. For example, if $F = (x_1 \vee \overline{x_2}) \wedge (\overline{x_1} \vee x_3)$, then $F' = (x_1 \vee y_2) \wedge (y_1 \vee x_3) \wedge (x_1 \vee y_1) \wedge (\overline{x_1} \vee \overline{y_1}) \wedge (x_2 \vee y_2) \wedge (\overline{x_2} \vee \overline{y_2})$.

It is easy to see that F' is a uniform CNF formula, and that F is satisfiable if and only if F' is satisfiable. This constitutes a reduction from SAT to Uniform-SAT, which concludes the proof. □

Theorem 2. SpecCon$(\beta + 1, \beta)$ *is* NP*-complete for any integer* $\beta \geq 2$*.*

Proof. The membership of SpecCon$(\beta + 1, \beta)$ in NP is clear. In what follows we reduce Uniform-SAT to SpecCon$(\beta + 1, \beta)$, which by Lemma 1 will prove the NP-completeness of the latter.

Let $c_1 \wedge c_2 \wedge \ldots \wedge c_m$ be an input to Uniform-SAT where c_j, $1 \leq j \leq m$, is a uniform clause. Assume the variable set is $\{x_1, x_2, \ldots, x_n\}$. We construct an instance of SpecCon$(\beta + 1, \beta)$ as follows.

- **Channels:** There are $\beta + 1$ channels $\{0, 1, 2, \ldots, \beta\}$.
- **SUs:** 1) For each variable x_i, there is a corresponding SU X_i with spectrum map SpecMap$(X_i) = \{0, 1, 2, \ldots, \beta\}$ (which contains all possible channels); 2) for each clause c_j, $1 \leq j \leq m$, there is a corresponding SU C_j with SpecMap$(C_j) = \{p_j\}$, where $p_j = 1$ if c_j is positive and $p_j = 0$ if c_j is

negative; 3) there is an SU Y_2 with $\textsc{SpecMap}(Y_2) = \{2\}$. For every $1 \leq i \leq n$ and $2 \leq k \leq \beta$, there is an SU $Y_{i,k}$ with $\textsc{SpecMap}(Y_{i,k}) = \{k\}$; and 4) all SUs have the same antenna budget β.

- **Potential Graph:** For each clause c_j and each variable x_i that appears in c_j (either as x_i or $\overline{x_i}$), there is a potential edge between X_i and C_j. For each $1 \leq i \leq n$ and $3 \leq k \leq \beta$, there is a potential edge between X_i and $Y_{i,k}$. Finally, there is a potential edge between Y_2 and every X_i, $1 \leq i \leq n$.

Denote the above cognitive radio network by $\mathcal{I}$, which is also an instance of $\textsc{SpecCon}(\beta + 1, \beta)$. We now prove that $c_1 \wedge c_2 \wedge \ldots \wedge c_m$ is satisfiable if and only if $\mathcal{I}$ is connectable.

First consider the "only if" direction. Let $A : \{x_1, \ldots, x_n\} \to \{0, 1\}$ be a satisfying assignment of $c_1 \wedge c_2 \wedge \ldots \wedge c_m$, where 0 stands for FALSE and 1 for TRUE. Define a spectrum assignment as follows. For each $1 \leq i \leq n$, let user X_i open the channels $\{2, 3, \ldots, \beta\} \cup \{A(i)\}$. Every other SU opens the only channel in its spectrum map.

We verify that $\mathcal{I}$ is connected under the above spectrum assignment. For each $1 \leq i \leq n$, X_i is connected to Y_2 through channel 2. Then, for every $2 \leq l \leq \beta$, $Y_{i,l}$ is connected to X_i through channel l. Now consider SU C_j where $1 \leq j \leq m$. Since A satisfies the clause c_j, there exists $1 \leq i \leq n$ such that: 1) x_i or $\overline{x_i}$ occurs in c_j; and 2) $A(x_i) = 1$ if c_j is positive, and $A(x_i) = 0$ if c_j is negative. Thus X_i and C_j are connected through channel $A(x_i)$. Therefore the realization graph is connected, completing the proof of the "only if" direction.

We next consider the "if" direction. Suppose there is a spectrum assignment that makes $\mathcal{I}$ connected. For every $1 \leq i \leq n$ and $2 \leq l \leq \beta$, X_i must open channel l, otherwise $Y_{i,l}$ will become an isolated vertex in the realization graph. Since X_i can open at most β channels in total, it can open at most one of the two remaining channels $\{0, 1\}$. We assume w.l.o.g. that X_i opens exactly one of them, which we denote by a_i.

Now, for the formula $c_1 \wedge c_2 \wedge \ldots \wedge c_m$, we define a truth assignment $A : \{x_1, \ldots, x_n\} \to \{0, 1\}$ as $A(x_i) = a_i$ for all $1 \leq i \leq n$. We show that A satisfies the formula. Fix $1 \leq j \leq m$ and assume that c_j is negative (the case where c_j is positive is totally similar). Since the spectrum map of SU C_j only contains channel 0, some of its neighbors must open channel 0. Hence, there exists $1 \leq i \leq n$ such that $\overline{x_i}$ appears in c_j and the corresponding SU X_i opens channel 0. By our construction of A, we have $A(x_i) = 0$, and thus the clause c_j is satisfied by A. Since j is chosen arbitrarily, the formula $c_1 \wedge c_2 \wedge \ldots \wedge c_m$ is satisfied by A. This completes the reduction from Uniform-SAT to $\textsc{SpecCon}(\beta, \beta + 1)$, and the theorem follows. □

Corollary 1. $\textsc{SpecCon}(k, \beta)$ *is* NP*-complete for any integers* $k > \beta \geq 2$.

Proof. By a simple reduction from $\textsc{SpecCon}(\beta + 1, \beta)$: Given an instance of $\textsc{SpecCon}(\beta+1, \beta)$, create $k-\beta-1$ new channels and add them to the spectrum map of an (arbitrary) SU. This gives a instance of $\textsc{SpecCon}(k, \beta)$. Since the new channels are only contained in one SU, they should not be opened, and thus the two instances are equivalent. Hence the theorem follows. □

Theorem 2 indicates that the SPECTRUM CONNECTIVITY problem is NP-complete even if the cognitive radio network only has three channels. We further strengthen this result by proving the following theorem:

Theorem 3. *The* SPECTRUM CONNECTIVITY *problem is* NP*-complete even if there are only two channels.*

Proof. We present a reduction from UNIFORM-SAT similar as in the proof of Theorem 2. Let $c_1 \wedge c_2 \wedge \ldots \wedge c_m$ be a uniform CNF clause with variable set $\{x_1, x_2, \ldots, x_n\}$. Construct a cognitive radio network as follows: There are two channels {0,1}. For each variable x_i there is a corresponding SU X_i with spectrum map $\text{SPECMAP}(X_i) = \{0, 1\}$ and antenna budget $\beta(X_i) = 1$. For each clause c_j there is a corresponding SU C_j with $\text{SPECMAP}(C_j) = \{p_j\}$ and $\beta(C_j) = 1$, where $p_j = 1$ if c_j is positive and $p_j = 0$ if c_j is negative. There is an SU Y with $\text{SPECMAP}(Y) = \{0, 1\}$ and $\beta(Y) = 2$. Note that, unlike in the case of $\text{SPECCON}(k, \beta)$, SUs can have different antenna budgets. Finally, the edges of the potential graph include: $\{X_i, C_j\}$ for all i, j such that x_i or $\overline{x_i}$ appears in c_j, and $\{Y, X_i\}$ for all i. This completes the construction of the cognitive radio network, which is denoted by $\mathcal{I}$. By an analogous argument as in the proof of Theorem 2, $c_1 \wedge c_2 \wedge \ldots \wedge c_m$ is satisfiable if and only if $\mathcal{I}$ is connectable, concluding the proof of Theorem 3. □

Theorem 3 is sharp in that, as noted before, the problem is polynomial-time solvable when there is only one channel.

3.2 Exact Algorithms

In this subsection we design algorithms for deciding whether a given cognitive radio network is connectable. Since the problem is NP-complete, we cannot expect a polynomial time algorithm.

Let n, k, t denote the number of SUs, the number of channels, and the maximum size of any SU's spectrum map, respectively ($t \leq k$). The simplest idea is to exhaustively examine all possible spectrum assignments to see if there exists one that makes the network connected. Since each SU can have at most 2^t possible ways of opening channels, the number of assignments is at most 2^{tn}. Checking each assignment takes $\text{poly}(n, k)$ time. Thus the running time of this approach is bounded by $2^{tn}(nk)^{O(1)}$, which is reasonable when t is small. However, since in general t can be as large as k, this only gives a $2^{O(kn)}$ bound, which is unsatisfactory if k is large. In the following we present another algorithm for the problem that runs faster than the above approach when k is large.

Theorem 4. *There is an algorithm that decides whether a given cognitive radio network is connectable in time* $2^{O(k+n \log n)}$*, where* n *and* k *are the number of SUs and channels respectively.*

Proof. Let $\mathcal{I}$ be a given cognitive radio network with potential graph $\mathcal{PG}$. Let n be the number of SUs and k the number of channels. Assume that $\mathcal{I}$ is connected under some spectrum assignment. Clearly the realization graph contains

a spanning tree of $\mathcal{PG}$, say T, as a subgraph. If we change the potential graph to T while keeping all other parameters unchanged, the resulting network will still be connected under the same spectrum assignment. Thus, it suffices to check whether there exists a spanning tree T of $\mathcal{G}$ such that $\mathcal{I}$ is connectable when substituting T for $\mathcal{PG}$ as its potential graph. Using the algorithm of [5], we can list all spanning trees of $\mathcal{PG}$ in time $O(Nn)$ where N is the number of spanning trees of $\mathcal{PG}$. By Cayley's formula [2, 11] we have $N \leq n^{n-2}$. Finally, for each spanning tree T, we can use the algorithm in Theorem 9 (which will appear in Section 5) to decide whether the network is connectable in time $2^{O(k)}n^{O(1)}$. The total running time of the algorithm is $O(n^{n-2})2^{O(k)}n^{O(1)} = 2^{O(k+n\log n)}$. □

Combining Theorem 4 with the brute-force approach, we obtain:

Corollary 2. *The* Spectrum Connectivity *problem is solvable can be solved in time* $2^{O(\min\{kn,k+n\log n\})}$*, with* n *and* k *being the number of SUs and channels respectively.*

4 Spectrum Connectivity with Complete Potential Graphs

In this section we consider the special case of the Spectrum Connectivity problem, in which the potential graph of the cognitive radio network is complete. We first show that this restriction does not make the problem tractable in polynomial time.

Theorem 5. *The* Spectrum Connectivity *problem is NP-complete even when the potential graph is complete and all SUs have the same antenna budget* $\beta = 2$.

Proof. The membership in NP is trivial. The hardness proof is by a reduction from the Hamiltonian Path problem, which is to decide whether a given graph contains a Hamiltonian path, i.e., a simple path that passes every vertex exactly once. The Hamiltonian Path problem is well-known to be NP-complete [6]. Let $G = (V, E)$ be an input graph of the Hamiltonian Path problem. Construct an instance of the Spectrum Connectivity problem as follows: The collection of channels is E and the set of SUs is V; that is, we identify a vertex in V as an SU and an edge in E as a channel. For every $v \in V$, the spectrum map of v is the set of edges incident to v. All SUs have antenna budget $\beta = 2$. Denote this cognitive radio network by $\mathcal{I}$. We will prove that G contains a Hamiltonian path if and only if $\mathcal{I}$ is connectable.

First suppose G contains a Hamiltonian path $P = v_1v_2 \dots v_n$, where $n = |V|$. Consider the following spectrum assignment of $\mathcal{I}$: for each $1 \leq i \leq n$, let SU v_i open the channels corresponding to the edges incident to v_i in the path P. Thus all SUs open two channels except for v_1 and v_n each of whom opens only one. For every $1 \leq i \leq n-1$, v_i and v_{i+1} are connected through the channel (edge) $\{v_i, v_{i+1}\}$. Hence the realization graph of $\mathcal{I}$ under this spectrum assignment is connected.

Now we prove the other direction. Assume that $\mathcal{I}$ is connectable. Fix a spectrum assignment under which the realization graph of $\mathcal{I}$ is connected, and consider this particular realization graph $\mathcal{RG} = (V, E')$. Let $\{v_i, v_j\}$ be an arbitrary edge in E'. By the definition of the realization graph, there is a channel opened by both v_i and v_j. Thus there is an edge in E incident to both v_i and v_j, which can only be $\{v_i, v_j\}$. Therefore $\{v_i, v_j\} \in E$. This indicates $E' \subseteq E$, and hence $\mathcal{RG}$ is a connected spanning subgraph of G. Since each SU can open at most two channels, the maximum degree of $\mathcal{RG}$ is at most 2. Therefore $\mathcal{RG}$ is either a Hamiltonian path of G, or a Hamiltonian cycle which contains a Hamiltonian path of G. Thus, G contains a Hamiltonian path.

The reduction is complete and the theorem follows. □

Notice that the reduction used in the proof of Theorem 5 creates a cognitive radio network with an unbounded number of channels. Thus Theorem 5 is not stronger than Theorem 1 or 3. Recall that Theorem 3 says the Spectrum Connectivity problem is NP-complete even if there are only two channels. In contrast we will show that, with complete potential graphs, the problem is polynomial-time tractable when the number of channels is small.

Theorem 6. *The* Spectrum Connectivity *problem with complete potential graphs can be solved in* $2^{2^k+O(k)}n^{O(1)}$ *time, where* k *is the number of channels and* n *is the number of SUs.*

Proof. Consider a cognitive radio network $\mathcal{I}$ with SU set U, channel set C and a complete potential graph, i.e., there is a potential edge between every pair of distinct SUs. Let $n = |U|$ and $k = |C|$. For each spectrum assignment $\mathcal{SA}$, we construct a corresponding *spectrum graph* $\mathcal{G}_{\mathcal{SA}} = (V, E)$ where $V = \{C' \subseteq C \mid \exists u \in U \text{ s.t. } \mathcal{SA}(u) = C'\}$ and $E = \{\{C_1, C_2\} \mid C_1, C_2 \in V; C_1 \cap C_2 \neq \emptyset\}$. Thus, V is the collection of subsets of C that is opened by some SU, and E reflexes the connectivity between pairs of SUs that open the corresponding channels. Since each vertex in V is a subset of C, we have $|V| \leq 2^k$, and the number of different spectrum graphs is at most 2^{2^k}.

We now present a relation between $\mathcal{G}_{\mathcal{SA}}$ and the realization graph of $\mathcal{I}$ under $\mathcal{SA}$. If we label each vertex u in the realization graph with $\mathcal{SA}(u)$, and contract all edges between vertices with the same label, then we obtain precisely the spectrum graph $\mathcal{G}_{\mathcal{SA}} = (V, E)$. Therefore, in the language of graph theory, $\mathcal{G}_{\mathcal{SA}} = (V, E)$ is a minor of the realization graph under $\mathcal{SA}$. Since graph minor preserves connectivity, $\mathcal{I}$ is connectable if and only if there exists a connected spectrum graph. Hence we can focus on the problem of deciding whether a connected spectrum graph exists.

Consider all possible graphs $G = (V, E)$ such that $V \subseteq 2^C$, and $E = \{\{C_1, C_2\} \mid C_1, C_2 \in V; C_1 \cap C_2 \neq \emptyset\}$. There are 2^{2^k} such graphs each of which has size $2^{O(k)}$. Thus we can list all such graphs in $2^{2^k+O(k)}$ time. For each graph G, we need to check whether it is the spectrum graph of some spectrum assignment of $\mathcal{I}$. We create a bipartite graph in which nodes on the left side are the SUs in $\mathcal{I}$, and nodes on the right side all the vertices of G. We add an edge

between an SU u and a vertex C' of G if and only if $C' \subseteq \textsc{SpecMap}(u)$ and $|C'| \leq \beta(u)$, that is, u can open C' in a spectrum assignment. The size of H is $\text{poly}(n, 2^k)$ and its construction can be finished in $\text{poly}(n, 2^k)$ time. Now, if G is the spectrum graph of some spectrum assignment $\mathcal{SA}$, then we can identify $\mathcal{SA}$ with a subgraph of H consisting of all edges $(u, \mathcal{SA}(u))$ where u is an SU. In addition, in this subgraph we have

- every SU u has degree exactly one; and
- every node C' on the right side of H has degree at least one.

Conversely, a subgraph of H satisfying the above two conditions clearly induces a spectrum assignment whose spectrum graph is exactly G. Therefore it suffices to examine whether H contains such a subgraph. Furthermore, the above conditions are easily seen to be equivalent to:

- every SU u has degree at least one in G; and
- G contains a *matching* that includes all nodes on the right side.

The first condition can be checked in time linear in the size of H, and the second one can be examined by any polynomial time algorithm for bipartite matching (e.g., [7]). Therefore, we can decide whether such subgraph exists (and find one if so) in time $\text{poly}(n, 2^k)$. By our previous analyses, this solves the Spectrum Connectivity problem with complete potential graphs. The total running time of our algorithm is $2^{2^k+O(k)}\text{poly}(n, 2^k) = 2^{2^k+O(k)}n^{O(1)}$. □

Theorem 7. Spectrum Connectivity *with complete potential graphs is fixed parameter tractable when parameterized by the number of channels.*

5 Trees as Potential Graphs

In this section, we study another special case of the Spectrum Connectivity problem where the potential graph of the cognitive radio network is a tree. Many NP-hard combinatorial problems become easy on trees, e.g., the dominating set problem and the vertex cover problem. Nonetheless, as indicated by the following theorem, the Spectrum Connectivity problem remains hard on trees.

Theorem 8. *The* Spectrum Connectivity *problem is* NP*-complete even if the potential graph is a tree of depth one.*

Proof. We give a reduction from the Vertex Cover problem which is well known to be NP-complete [6]. Given a graph $G = (V, E)$ and an integer r, the Vertex Cover problem is to decide whether there exists r vertices in V that cover all the edges in E. Construct a cognitive radio network $\mathcal{I}$ as follows. The set of channels is $C = \{c_v \mid v \in V\}$. For each edge $e = \{u, v\} \in E$ there is an SU U_e with $\textsc{SpecMap}(U_e) = \{c_u, c_v\}$ and antenna budget 2. There is another SU M with $\textsc{SpecMap}(M) = C$ and antenna budget r. The potential graph is a star centered at M, that is, there is a potential edge between M and U_e for every $e \in E$. This finishes the construction of $\mathcal{I}$.

We prove that G has a vertex cover of size r if and only if $\mathcal{I}$ is connectable. First assume G has a vertex cover $S \subseteq V$ with $|S| \leq r$. Define a spectrum assignment $A(S)$ as follows: M opens the channels $\{c_v \mid v \in S\}$, and U_e opens both channels in its spectrum map for all $e \in E$. Since S is a vertex cover, we have $u \in S$ or $v \in S$ for each $e = \{u, v\} \in E$. Thus at least one of c_u and c_v is opened by M, which makes it connected to U_e. Hence the realization graph is connected. On the other hand, assume that the realization graph is connected under some spectrum assignment. For each $e = \{u, v\} \in E$, since the potential edge $\{M, U_e\}$ is realized, M opens at least one of c_u and c_v. Now define $S = \{v \in V \mid c_v \text{ is opened by } M\}$. It is clear that S is a vertex cover of G of size at most $\beta(M) = r$. This completes the reduction, and the theorem follows. □

We next show that, in contrast to Theorems 2 and 3, this special case of the problem is polynomial-time solvable when the number of channels is small.

Theorem 9. *Given a cognitive radio network whose potential graph is a tree, we can check whether it is connectable in $2^{O(t)}(kn)^{O(1)}$ time, where n is the number of SUs, k is the number of channels, and t is the maximum size of any SU's spectrum map. In particular, this running time is at most $2^{O(k)}n^{O(1)}$.*

Proof. Let $\mathcal{I}$ be a given cognitive radio network whose potential graph $\mathcal{PG} = (V, E)$ is a tree. Root $\mathcal{PG}$ at an arbitrary node, say r. For each $v \in V$ let $\mathcal{PG}_v$ denote the subtree rooted at v, and let $\mathcal{I}_v$ denote the cognitive radio network obtained by restricting $\mathcal{I}$ on $\mathcal{PG}_v$. For every subset $S \subseteq \textsc{SpecMap}(v)$, define $f(v, S)$ to be 1 if there exists a spectrum assignment that makes $\mathcal{I}_v$ connected in which the set of channels opened by v is exactly S; let $f(v, S) = 0$ otherwise. For each channel $c \in C$, define $g(v, c)$ to be 1 if there exists S, $\{c\} \subseteq S \subseteq \textsc{SpecMap}(v)$, for which $f(v, S) = 1$; define $g(v, c) = 0$ otherwise. Clearly $\mathcal{I}$ is connectable if and only if there exists $S \subseteq \textsc{SpecMap}(r)$ such that $f(r, S) = 1$.

We compute all $f(v, S)$ and $g(v, c)$ by dynamic programming in a bottom-up manner. Initially all values to set to 0. The values for leaf nodes are easy to obtain. Assume we want to compute $f(v, S)$, given that the values of $f(v', S')$ and $g(v', c)$ are all known if v' is a child of v. Then $f(v, S) = 1$ if and only if for every child v' of v, there exists $c \in S$ such that $g(v', c) = 1$ (in which case v and v' are connected through channel c). If $f(v, S)$ turns out to be 1, we set $g(v, c)$ to 1 for all $c \in S$. It is easy to see that $g(v, c)$ will be correctly computed after the values of $f(v, S)$ are obtained for all possible S. After all values have been computed, we check whether $f(r, S) = 1$ for some $S \subseteq \textsc{SpecMap}(r)$.

Denote $n = |V|$, $k = |C|$, and $t = \max_{v \in V} |\textsc{SpecMap}(v)|$. There are at most $n(2^t + k)$ terms to be computed, each of which takes time $\text{poly}(n, k)$ by our previous analysis. The final checking step takes $2^t\text{poly}(n, k)$ time. Hence the total running time is $2^t\text{poly}(n, k) = 2^t(kn)^{O(1)}$, which is at most $2^{O(k)}n^{O(1)}$ since $t \leq k$. Finally note that it is easy to modify the algorithm so that, given a connectable network it will return a spectrum assignment that makes it connected. □

Corollary 3. Spectrum Connectivity *with trees as potential graphs is fixed parameter tractable when parameterized by the number of channels.*

6 Conclusion

In this paper, we systematically study the algorithmic complexity of connectivity problem in cognitive radio networks through spectrum assignment. The hardness of the problem in the general case and several special cases are addressed. Our work gives a better understanding of the complexity of the problem. Exact algorithms are also derived to check whether the network is connectable. Due to interference, the connected nodes can not communicate simultaneously. One meaningful extension of this work is how to schedule the links such that the network throughput is optimized under realistic interference models. Another future work is to design efficient distributed channel assignment algorithms to achieve network connectivity.

Acknowledgements. The authors would like to give thanks to Dr. Thomas Moscibroda at Microsoft Research Asia for his introduction of the original problem.

References

1. Bahl, P., Chandra, R., Moscibroda, T., Murty, R., Welsh, M.: White space networking with Wi-Fi like connectivity. In: ACM SIGCOMM (2009)
2. Cayley, A.: A theorem on trees. Quarterly Journal on Pure and Applied Mathematics 23, 376–378 (1889)
3. Dolev, S., Gilbert, S., Guerraoui, R., Newport, C.: Gossiping in a Multi-channel Radio Network. In: Pelc, A. (ed.) DISC 2007. LNCS, vol. 4731, pp. 208–222. Springer, Heidelberg (2007)
4. Downey, R.G., Fellows, M.R.: Parameterized Complexity. Springer (1998)
5. Gabow, H.N., Myers, E.W.: Finding all spanning trees of directed and undirected graphs. SIAM J. Comput. 7(3), 280–287 (1978)
6. Garey, M.R., Johnson, D.S.: Computers and Intractability: A Guide to the Theory of NP-Completeness. W.H. Freeman (1979)
7. Hopcroft, J.E., Karp, R.M.: An $n^{5/2}$ algorithm for maximum matchings in bipartite graphs. SIAM J. Comput. 2(4), 225–231 (1973)
8. Lu, D., Huang, X., Li, P., Fan, J.: Connectivity of large-scale cognitive radio ad hoc networks. In: IEEE INFOCOM (2012)
9. Ren, W., Zhao, Q., Swami, A.: Connectivity of cognitive radio networks: Proximity vs. opportunity. In: ACM CoRoNet (2009)
10. Ren, W., Zhao, Q., Swami, A.: Power control in cognitive radio networks: How to cross a multi-lane highway. IEEE JSAC 27(7), 1283–1296 (2009)
11. Shukla, A.B.: A short proof of Cayley's tree formula. arXiv:0908.2324v2
12. Wang, B., Liu, K.J.R.: Advances in cognitive radio networks: A survey. IEEE J. Selected Topics in Signal Processing 5(1), 5–23 (2011)
13. Xu, C., Huang, J.: Spatial spectrum access game: nash equilibria and distributed learning. In: ACM MobiHoc (2012)
14. Yuan, Y., Bahl, P., Chandra, R., Moscibroda, T., Wu, Y.: Allocating dynamic time-spectrum blocks in cognitive radio networks. In: ACM MobiCom (2007)
15. Zhou, X., Gandhi, S., Suri, S., Zheng, H.: ebay in the sky: Strategyproof wireless spectrum auctions. In: ACM MobiCom (2008)

On Some Bounds on the Optimum Schedule Length in the SINR Model

Tigran Tonoyan

TCS Sensor Lab, Centre Universitaire d'Informatique
Route de Drize 7, 1227 Carouge, Geneva, Switzerland
tigran.tonoyan@unige.ch

Abstract. In this paper the problem of wireless transmissions scheduling in the path-loss/SINR model is considered for different fixed power schemes. The lower bounds for the optimum schedule length proven by Kesselheim and Voecking are considered. It is shown that the lower bound for the linear power scheme is tight, by presenting a constant-factor approximation algorithm. On the other hand, it is shown that the lower bounds proven for many other interesting power schemes can be essentially far from the optima.

1 Introduction

In the problem of wireless transmissions scheduling a set of links (transmission requests between sender and receiver nodes) is given and the goal is to schedule those transmissions into the minimum number of time slots subject to the signal interference. The problem has been considered in several models such as the models based on the notion of a *communication graph* and the *path-loss model*. In the path-loss model the signal power decreases with the distance proportionally to a negative power of the distance. Graph-based models are actually approximations of the path-loss model (they reflect the case of a very high path loss) and are easier to analyze from the theoretical perspective. However, during the recent decade the interest in algorithms in the path-loss model increased because it has been shown that for some instances of the scheduling problem the optimum solution in the path-loss model yields essentially better schedules (smaller number of time slots) than the optimum solution in the graph-based models [16], [6]. This could be explained by the fact that the graph-based models do not take into the account the accumulative nature of wireless signal interference.

In the path-loss model the interference of *all* concurrently transmitting nodes is summed and this sum is used in the *SINR (Signal to Interference plus Noise Ratio)* formula to describe if the signal sent is recovered by the receiver or not. The formula reflects the fact that in order to have a successful transmission one needs to have a receive signal power that dominates the interference by all concurrent transmissions plus the ambient noise.

The problem of scheduling has been considered in different settings. For example, depending on the technical capabilities of the wireless nodes, one can

A. Bar-Noy and M.M. Halldórsson (Eds.): ALGOSENSORS 2012, LNCS 7718, pp. 120–131, 2013.

optimize also the transmitting power levels of the nodes in order to schedule the transmissions into smaller number of slots. However in many cases the transmission power level is fixed and cannot be varied for scheduling purposes. In this case the scheduling problem is stated for fixed power schemes. In many cases the power schemes are considered as functions of link-length (the distance between sender and receiver nodes). This can be explained by the fact that in the distributed setting nodes have to choose such power schemes, which require only a local knowledge (depend only on the link-length). Examples of commonly considered power schemes are the *uniform* and the *linear (min-energy)* power schemes. When the uniform scheme is used, the transmit signal power is constant (is the same for all links) and when the linear scheme is used, the receive signal power is constant (the signal power at a receiver node that was sent by the corresponding sender). The linear power scheme is the main subject of this paper. It is the minimum power scheme to provide a constant SINR at the receiver without other nodes transmitting, hence the node wastes a minimum energy on the transmission when using the linear power scheme.

The main result of this paper is a constant factor approximation algorithm for the scheduling problem with the linear power scheme. We show that the upper bound provided by the algorithm matches the lower bounds presented in [13] and [3] (up to constant factors). We also consider similar lower bounds proven in [13] for other power schemes and show that those can be very far from the optimum.

1.1 Related Work

The algorithmic/analytical treatment of the scheduling problem has begun relatively recently. Before the majority of the work was based on heuristics (see [15] for an overview) and the algorithms that were designed did not have theoretically proven performance guarantees. One of the first attempts to design approximation algorithms for networks with arbitrary topology was done in [15]. They consider scheduling together with power control. An algorithm is designed to work for any network topology on the plane, however the approximation factor depends on the network topology and can be linear in the number of links. It is also shown in [15] that for some network instances many of the heuristics designed prior to their work give exponentially worse results than their scheduling algorithm. In [2] a scheduling algorithm is designed for the linear power scheme in the framework of cross-layer optimization and further developed for linear and uniform power schemes in [1]. In both cases the approximation ratio depends poly-logarithmically on a parameter Δ which is the maximum to minimum link-length ratio and can grow linearly with the number of links. In [5] it is shown that the problem of scheduling with the uniform power scheme is an NP-hard problem and also an $O(\log \Delta)$-approximation algorithm is presented. In [4] an $O(\log n)$-approximation algorithm is presented for the uniform power scheme, where n is the number of links. The main part of the algorithm consists of a subroutine that finds a constant factor approximation for *one-shot scheduling* (also referred as *capacity problem*), that is finding a feasible set of

links of maximum cardinality. Scheduling is done by iteratively calling the subroutine and removing the resulting set. In [10] this algorithm is further refined to work in a more general setting and to find a constant factor approximation for *k-throughput problem* (finding the maximum cardinality subset that can be scheduled into k slots). Several other papers (e.g. [9], [12]) considered the problem of one-shot scheduling with different power schemes (i.e. the class of *sub-linear* power schemes; see Section 3 for the definition) and with power control and obtained constant factor approximation algorithms for those cases. This in turn led to $O(\log n)$-approximation for those variants of the scheduling problem. Another set of papers [3], [13], [8] considered some algorithms derived from the ALOHA protocol. In [3] a randomized algorithm is presented for the linear power scheme that guarantees $O(OPT + \log^2 n)$-approximation, where OPT is the optimum schedule length. In [13] a similar algorithm is proven to work for all sub-linear power schemes and provide $O(\log^2 n)$-approximation. In [8] it is proven that the algorithm from [13] actually provides $O(\log n)$-approximation.

This paper is a revised version of the manuscript [19] which has been presented in the workshop WRAWN 2010.

2 Problem Definition

Throughout this work we assume that the nodes of a wireless network are represented by points in an Euclidean metric space X with a distance function d. The ball in X with the center at p and radius $r > 0$ is the set

$$B(p, r) = \{q \in X | d(p, q) < r\}.$$

Let μ be the standard Lebesgue measure: then for any two balls A and B with radii a and b respectively,

$$\frac{\mu(A)}{\mu(B)} = \left(\frac{a}{b}\right)^m, \tag{1}$$

where $m \geq 1$ is the dimension of X. These are the only properties of Euclidean space that we use (as opposed to any other metric measure space).

Assume further a set of links $\Gamma = \{1, 2, \ldots, n\}$ is given, where each link v represents a communication request from a sender node s_v to a receiver node r_v. For any $v, w \in \Gamma$ let us denote $d_{vw} = d(s_v, r_w)$ and $l_v = d(s_v, r_v)$. l_v is called *the length* of the link v. *A power scheme* P is a rule of assigning a power level of transmission to each link (more precisely, to the sender node), so for a given set of links Γ it yields a function $P : \Gamma \to R_+$. We adopt the path-loss model of transmission, where the received signal power from the sender of w at the receiver of v is $P(w)/d_{wv}^{\alpha}$ and α is a constant called *path-loss exponent* and typically ranges between 2 and 6 (see [17] for details). Throughout this paper we assume that $\alpha > m$ (so for the Euclidean plane we assume $\alpha > 2$). The signal interference is calculated according to the SINR model, where the transmission of a link v is successful if and only if the following condition holds:

$$SINR(S, v) = P(v)/l_v^{\alpha} / \left(\sum_{w \in S \setminus v} P(w)/d_{wv}^{\alpha} + N \right) \geq \beta, \tag{2}$$

where S is the set of concurrently scheduled links, $N \geq 0$ is the ambient noise and $\beta \geq 1$ is the minimum SINR required for a message to be successfully received. If (2) is satisfied for each link $v \in S$, we say that S is *SINR-feasible* (or just feasible) with respect to P.

A schedule for a set Γ with power scheme P is a collection of disjoint subsets or *slots* $\{S_1, S_2, \dots\}$ of Γ, where each S_i is feasible with respect to P. The number of slots is called *the length* of the schedule.

In the *scheduling problem* we are given a set of links Γ and a power scheme P and the goal is to find a schedule (with respect to P) of minimum length for Γ. The length of such a schedule is denoted $OPT_P(\Gamma)$.

2.1 Affectance and Noise

As it is noted in [18], if for a constant $c > 1$ it holds that $P(v) \geq c\beta N l_v^\alpha$ for all $v \in \Gamma$ (i.e. the power levels of nodes are a factor of c higher than the minimum possible non-trivial power level), then setting the noise factor N to 0 will affect the resulting schedule lengths only by a constant factor (depending on c), so we make this assumption e.g. for $c = 2$ and let $N = 0$ (the reader might notice that this assumption will not be needed for the linear power scheme).

For a given power scheme P, we will use the notion of *affectance* of a link v by a set S (we use the variant used in [13], see also [10] and [4]):

$$A_P(S, v) = \sum_{w \in S \setminus v} \min\left\{\frac{P(w) l_v^\alpha}{P(v) d_{wv}^\alpha}, 1\right\}.$$

Since we assume that $\beta \geq 1$ and $N = 0$, it can be verified that $A_P(S, v) \leq 1/\beta$ if and only if $SINR(S, v) \geq \beta$. Note also that the affectance has the useful property of additivity: if $S \cap T = \emptyset$ then $A_P(S \cup T, v) = A_P(S, v) + A_P(T, v)$. Hence from now on we will use "$A_P(S, v) \leq 1/\beta$ for all $v \in S$" as the criterion of feasibility.

3 Sub-Linear and Super-Linear Power Schemes

The following are power schemes that are commonly considered in the literature: the *uniform* power scheme U ($U(v) = const$ for all v), the *linear* power scheme L ($L(v) = c l_v^\alpha$ for all v, where $c > 0$ is constant) and the *mean* or *square-root* power scheme M ($M(v) = c l_v^{\alpha/2}$ for all v, where $c > 0$ is constant). All of them belong to the class of sub-linear power schemes, which we define as follows.

Let us first consider the following partial order '$\preceq$' on the set of power schemes. For two power schemes P and Q we write $P \preceq Q$ iff P grows slower than Q when considered as a function of link-length, i.e. for any two links v, w such that $l_v \geq l_w$ it holds that $P(v)/P(w) \leq Q(v)/Q(w)$.

We say that a power scheme P belongs to the class of *sub-linear* power schemes if $P \preceq L$. On the other hand, we say that P belongs to the class of super-linear power schemes if $L \preceq P$. Note that we include L in both of these classes. This is because (as we will see afterwards) it shares common properties with other power schemes from these classes.

3.1 Lower Bounds

The following lower bound is proven in [13]. First consider the following lemma.

Lemma 1. *[13] Given a set Γ of links and another link v with $l_v \leq l_w$ for all $w \in \Gamma$, then for any sub-linear power scheme P,*

$$OPT_P(\Gamma) = \Omega(A_P(\Gamma, v)).$$

Let $S[v^+] = \{w \in S \setminus \{v\} : l_w \geq l_v\}$ for any set of links S. Then it follows from the lemma that for each sub-linear power scheme P and a set of links Γ the following is true:

$$OPT_P(\Gamma) = \Omega\left(\max_{v \in \Gamma} A_P(\Gamma[v^+], v)\right). \tag{3}$$

We note here that the symmetric analogue of Lemma 1 holds also for super-linear power schemes.

Lemma 2. *Given a set Γ of links and a link v with $l_v \geq l_w$ for all $w \in \Gamma$, then for any super-linear power scheme P,*

$$OPT_P(\Gamma) = \Omega(A_P(\Gamma, v)).$$

The proof of Lemma 2 is very similar to the proof of Lemma 1 and is omitted (see Lemma 7 in [13]).

Now let $S[v^-] = \{w \in S \setminus \{v\} : l_w \leq l_v\}$ for any set of links S. Then for each super-linear power scheme P and a set of links Γ it holds that

$$OPT_P(\Gamma) = \Omega\left(\max_{v \in \Gamma} A_P(\Gamma[v^-], v)\right). \tag{4}$$

Recall that the linear power scheme is both sub-linear and super-linear, hence both lower bounds hold for L. It follows then, that their sum is also a lower bound for L.

$$OPT_L(\Gamma) = \Omega\left(\max_{v \in \Gamma} A_L(\Gamma[v^-], v) + \max_{v \in \Gamma} A_L(\Gamma[v^+], v)\right). \tag{5}$$

Note that using additivity of affectance it can be shown that (5) is equivalent to the following lower bound proven in [3]:

$$OPT_L(\Gamma) = \Omega\left(\max_{v \in \Gamma} A_L(\Gamma, v)\right).$$

In the rest of the paper we try to find out how close are these lower bounds to the respective optima.

4 Algorithm Decreasing First-Fit

In this section we discuss an algorithm scheme that was inspired by [10]. For any power scheme P consider the first-fit algorithm that picks the links according to the decreasing order of the link-length then adds each next link v to the first slot S such that

$$A_P(S, v) \leq \gamma^{-\alpha}$$

for a certain constant $\gamma > 1$. We call this algorithm scheme, which is parameterized by P, Decreasing First-Fit(P).

Algorithm 1. Decreasing First-Fit(P)

1. Input: the set of links Γ
2. Consider empty slots S_i, $i = 1, 2, \ldots$
3. Process the links according to the decreasing order of length
   3.1 Add the current link v to the first slot S_i, s. t. $A_P(S_i, v) \leq \gamma^{-\alpha}$
4. Output the collection of non-empty slots $S_1, S_2, \ldots$

Lemma 3. *If t is the number of subsets given on the output of Decreasing First-Fit(P) then*

$$t = O(\max_{v \in \Gamma} A_P(\Gamma[v^+], v)).$$

Proof. Suppose $S_1, S_2, \ldots, S_t$ is the output of the algorithm. Let v be a link from S_t. By the definition of the algorithm we have

$$A_P(S_i[v^+], v) > \frac{1}{\gamma^\alpha} \text{ for all } i < t.$$

Since A is additive, then we have

$$A_P(\Gamma[v^+], v) = \sum_{i=1}^{t} A_P(S_i[v^+], v) > (t-1)/\gamma^\alpha.$$

On the other hand we have

$$A_P(\Gamma[v^+], v) \leq \max_{v \in \Gamma} A_P(\Gamma[v^+], v), \text{ so}$$

$$t < \gamma^\alpha \max_{v \in \Gamma} A_P(\Gamma[v^+], v) + 1$$

which proves the lemma, since it is assumed that γ is a constant. □

Note that the number of slots output by the algorithm matches the lower bound (3), so the only thing that remains to show in order to get a constant factor approximation for the scheduling problem is that the output of the algorithm is a feasible schedule. Let us show that this is true for the linear power scheme.

4.1 Decreasing First-Fit with the Linear Power Scheme

In this section we show that Decreasing First-Fit(L) is a constant factor approximation algorithm for scheduling with the linear power scheme. We prove that for a suitably chosen constant γ the output of the algorithm is a feasible schedule.

Correctness of the Algorithm. The following lemma is the main technical step towards proving that the result of the algorithm is a feasible schedule.

Lemma 4. *There is a constant $\gamma_0 = \gamma_0(\alpha, m)$ such that for any set of links S assigned to the same slot and $v \in S$ the following holds:*

$$A_L(S[v^-], v) \leq \frac{\gamma_0}{(\gamma - 2)^\alpha}.$$

Proof. (Outline) We use an area argument similar to the one presented in [10]. First it can be shown that the selection condition of the algorithm ensures that the balls $B(s_w, (\gamma - 1)l_w/2)$ for different s_w with $w \in S$ do not intersect, i.e. the links are spatially separated. Now the main idea is to split the metric space into concentric rings R_i for $i = 0, 1, \ldots$ centered at the sender node s_v and each having width $(\gamma - 1)l_v$. This will induce a partition of $S[v^-]$ into subsets $S_i[v^-] = \{w \in S[v^-] : s_w \in R_i\}$ for $i = 0, 1, 2, \ldots$. Let us consider the set of links in a partition $S_i[v^-]$. As it was noted above, the balls $B(s_w, (\gamma - 1)l_w/2)$ corresponding to these links do not intersect each other. On the other hand it can be argued that since $l_w \leq l_v$ for all these links, the balls are contained in a ring resulting from R_i by extending it in both directions by $(\gamma - 1)l_v/2$.

Now the fact that the sum of volumes of these balls is not more than the volume of the extended ring (due to countable additivity of the Lebesgue measure) can be expressed numerically as follows (after a number of simplifications):

$$\sum_{w \in S_i[v^-]} l_w^m \leq c l_v^m i^{m-1},$$

where $c > 0$ is a constant. Next step is to relate this inequality to affectance. Since the senders in $S_i[v^-]$ are roughly at a distance $(\gamma - 2)il_v$ from the receiver r_v, the affectance $A_L(S_i[v^-], v)$ can be bounded from above as follows:

$$A_L(S_i[v^-], v) \leq \frac{\sum_{w \in S_i[v^-]} l_w^\alpha}{((\gamma - 2)il_v)^\alpha} < \frac{\left(\sum_{w \in S_i[v^-]} l_w^m\right)^{\alpha/m}}{((\gamma - 2)il_v)^\alpha} \leq \frac{c'}{(\gamma - 2)^\alpha i^{\alpha/m}},$$

where the second inequality follows by using a known inequality for sums of positive real numbers. Since we assumed that $\alpha > m$, it follows that the partial affectances $A_L(S_i[v^-], v)$ fade away according to the law $i^{-\alpha/m}$, so the overall affectance $A_L(S[v^-], v)$ can be bounded by using the additivity property as follows:

$$A_L(S[v^-], v) \leq \frac{c'}{(\gamma - 2)^\alpha} \sum_{i=1}^{\infty} \frac{1}{i^{\alpha/m}} \leq \frac{\gamma_0}{(\gamma - 2)^\alpha},$$

for an appropriately chosen constant γ_0. □

Lemma 4 helps to prove the following theorem.

Theorem 1. *If* $\gamma \geq \sqrt[\alpha]{\beta\,(\gamma_0+1)} + 2$, *then the output of the algorithm is a feasible schedule.*

Proof. Let v be any link that is scheduled in a slot S. Consider all the links in S, which are longer than v. From the definition of the algorithm it is clear, that the affectance of v by these links is in total no more than $1/\gamma^\alpha$. According to Lemma 4, the affectance of v by the links shorter than v is less than $\gamma_0/(\gamma-2)^\alpha$, so the overall affectance of the link v is

$$A_L(S,v) < \frac{1}{\gamma^\alpha} + \frac{\gamma_0}{(\gamma-2)^\alpha} \leq \frac{\gamma_0+1}{(\gamma-2)^\alpha} \leq \frac{1}{\beta}.$$

□

The Approximation Ratio. The following theorem follows from Lemma 3, the bound (3) and Theorem 1.

Theorem 2. *If the condition of Theorem 1 is satisfied and* $\alpha > m$, *then Decreasing First-Fit(*L*) is a constant factor approximation algorithm for scheduling with the linear power scheme. Moreover, the following expression for the optimal schedule length holds true:*

$$OPT_L(\Gamma) = \Theta\left(\max_{v\in\Gamma} A_L(\Gamma[v^+], v)\right).$$

Because of (5) the following expression is also true for the optimum schedule length.

Corollary 1. *If* $\alpha > m$ *then*

$$OPT_L(\Gamma) = \Theta\left(\max_{v\in\Gamma} A_L(\Gamma[v^-], v) + \max_{v\in\Gamma} A_L(\Gamma[v^+], v)\right) = \Theta\left(\max_{v\in\Gamma} A_L(\Gamma, v)\right).$$

In fact the expression $OPT_L(\Gamma) = \Theta\left(\max_{v\in\Gamma} A_L(\Gamma, v)\right)$ could be proven directly, using an analogue of Lemma 3 (with the help of the same Decreasing First-Fit(L)). A possibly useful information that follows from this is that Decreasing First-Fit(L) would yield a constant factor approximation for the linear power scheme even if we considered the links in any order (not necessarily by decreasing link-length), provided that the output of the algorithm was a feasible schedule.

4.2 First-Fit for Other Super-Linear Power Schemes

In this section we note that Decreasing First-Fit(P) also works for other super-linear power schemes. The proof can be modeled using similar arguments as in the case of the linear power scheme, but there is yet another way of showing that.

First let us state an easy fact that follows from the definition of the partial order $\preceq$.

Lemma 5. *Consider power schemes P and Q such that $P \preceq Q$. Then for any set S and a link v it holds that*

$$A_Q(S[v^-], v) \leq A_P(S[v^-], v) \text{ and } A_P(S[v^+], v) \leq A_Q(S[v^+], v).$$

Let us fix any super-linear power scheme P_0. Let $\{S_1, S_2, \dots S_t\}$ be the result of Decreasing First-Fit(P_0). According to the algorithm statement,

$$A_{P_0}(S_i[v^+], v) < \gamma^{-\alpha} \text{ for all } v \in S_i \text{ and } i = 1, 2, ..., t.$$

Fix some i and $v \in S_i$. Then it follows from Lemma 5 that

$$A_L(S_i[v^+], v) \leq A_{P_0}(S_i[v^+], v) < \gamma^{-\alpha}$$

that is the conditions for Lemma 4 hold, which means there is a constant γ_0 such that $A_L(S_i[v^-], v) \leq \frac{\gamma_0}{(\gamma-2)^\alpha}$. But Lemma 5 implies that

$$A_{P_0}(S_i[v^-], v) \leq A_L(S_i[v^-], v) \leq \frac{\gamma_0}{(\gamma-2)^\alpha}.$$

Thus we have the following theorem.

Theorem 3. *If $\gamma \geq \sqrt[\alpha]{\beta\,(\gamma_0 + 1)} + 2$ then the output of Decreasing First-Fit(P) is a feasible schedule for any P, such that $L \preceq P$.*

This theorem and Lemma 3 together imply the following upper bound on the optimal schedule lengths.

Corollary 2. *For any P with $L \preceq P$, $OPT_P(\Gamma) = O\left(\max_{v \in \Gamma} A_P(\Gamma[v^+], v)\right)$.*

5 The Lower Bounds for Non-Linear Power Schemes

As we saw the lower bounds (3) and (5) are tight for the linear power scheme. Hence it is natural to ask the question how close are the lower bounds presented in Section 3 to the optima for non-linear power schemes. In this section we consider this question for power schemes P with $P \preceq L$ or $L \preceq P$.

First consider the power schemes P such that $U \preceq P \preceq L$. For any constant ϵ such that $0 < \epsilon < \alpha$ let $L_{-\epsilon}$ denote the power scheme which is given by formula $L_{-\epsilon}(v) = cl_v^{\alpha-\epsilon}$.

Theorem 4. *Consider any $\epsilon \in (0, \alpha)$ and a power scheme P such that $U \preceq P \preceq L_{-\epsilon}$. Then the bound (3) for P can be as far from the optimum as by a factor of $\Omega(n)$, where n is the number of links. Also for such a P Decreasing First-Fit(P) does not produce a feasible schedule for any constant value of γ.*

Proof. Let LG denote the power scheme given by the expression $LG(v) = cl_v^\alpha / \log l_v$. It has been shown in [7] (see Theorem 4.13 in [7]) that for each $\epsilon \in (0, \alpha)$ and for any power scheme P with $U \preceq P \preceq L_{-\epsilon}$ there is a family of networks Γ_ϵ such that $OPT_P(\Gamma_\epsilon) = \Omega(OPT_{LG}(\Gamma_\epsilon) \log\log \Delta)$ where $\Delta > 0$ is the

maximum link-length and in this case $\log\log\Delta = \Omega(|\Gamma_\epsilon|) = \Omega(n)$. This means that there is a schedule for Γ_ϵ with a constant number of slots that uses LG, but each schedule that uses P has to use $\Omega(n)$ slots. On the other hand, obviously $P \preceq L_{-\epsilon} \preceq LG$, so by applying Lemma 5 and using the lower bound (3) we get that

$$\max_{v\in\Gamma_\epsilon} A_P(\Gamma_\epsilon[v^+], v) \leq \max_{v\in\Gamma_\epsilon} A_{LG}(\Gamma_\epsilon[v^+], v) \in O(1),$$

which implies that $\max_{v\in\Gamma_\epsilon} A_P(\Gamma_\epsilon[v^+], v) \in O(1)$, while as it was noted above $OPT_P(\Gamma_\epsilon) \in \Omega(n)$. □

The theorem above is true in particular for U and M and many other sub-linear power schemes.

Now consider the power schemes P such that $P \preceq U$. Recall that the *capacity* problem is stated as follows: given the set of links Γ, find a feasible subset of maximum cardinality. Let $OPTC_P(\Gamma)$ denote the cardinality of such a subset for a set of links Γ and a power scheme P. It has been shown in [18] that for any set Γ and any power scheme P such that $P \preceq U$ or $L \preceq P$ it holds that

$$OPTC_L(\Gamma) = \Theta(OPTC_U(\Gamma)) = \Omega(OPTC_P(\Gamma)). \tag{6}$$

Using (6) and a simple technical argument we arrive to the following lemma.

Lemma 6. *Let Γ be a set of links and the power scheme P be such that $P \preceq U$ or $L \preceq P$. Then*

$$OPT_U(\Gamma) = O(OPT_P(\Gamma)\log n) \text{ and } OPT_L(\Gamma) = O(OPT_P(\Gamma)\log n),$$

where $n = |\Gamma|$.

Having Lemma 6, an argument similar to the one used for Theorem 4 can be applied to prove the following result concerning power schemes P with $P \preceq U$.

Theorem 5. *For any power scheme P with $P \preceq U$ the lower bound (3) is not tight and can be as far from the optimum as by a factor of $\Omega(n/\log n)$, where n is the number of links.*

It can be proven in a similar fashion that for power schemes P with $L \preceq P$ the lower bounds (4) can be very far from the optimum. In order to obtain a result similar to Theorem 4 we will use the following lemma. For any constant ϵ such that $\epsilon > 0$ let $L_{+\epsilon}$ denote the power scheme which is given by the expression $L_{+\epsilon}(v) = cl_v^{\alpha+\epsilon}$.

Lemma 7. *Consider any $\epsilon > 0$ and a power scheme P such that $L_{+\epsilon} \preceq P$. Then there is a network family Γ_ϵ with the following property: there is a schedule for Γ_ϵ that uses L and has a number of slots bounded by a constant, while each schedule that uses P must use $\Omega(n)$ slots.*

Theorem 6. *Consider any $\epsilon > 0$ and a power scheme P such that $L_{+\epsilon} \preceq P$. Then the bound (4) for P can be as far from the optimum as by a factor of $\Omega(n)$, where n is the number of links.*

Proof. Consider a network instance Γ_ϵ as in Lemma 7, i.e. the following properties hold: $OPT_L(\Gamma_\epsilon) \in O(1)$ and $OPT_P(\Gamma_\epsilon) \in \Omega(n)$. Since $L \preceq P$, then using Lemma 5 and (4) we have

$$\max_{v \in \Gamma_\epsilon} A_P(\Gamma_\epsilon[v^-], v) \leq \max_{v \in \Gamma_\epsilon} A_L(\Gamma_\epsilon[v^-], v) \in O(OPT_L(\Gamma_\epsilon)) = O(1),$$

so $OPT_P(\Gamma_\epsilon) \in \Omega(n \max_{v \in \Gamma_\epsilon} A_P(\Gamma_\epsilon[v^-], v))$. □

6 Conclusion

We considered the lower bounds (3) for sub-linear power schemes and (4) for super-linear power schemes. We showed that

1. (3) and (5) are tight for the linear power scheme and are achieved by the algorithm Decreasing First-Fit(L).
2. For all power schemes P with $P \preceq L_{-\epsilon}$ (where $\epsilon \in (0, \alpha)$) the bounds (3) are not tight and can be very far (up to a factor of $\Omega(n/\log n)$) from the corresponding optima.
3. For power schemes P with $L \preceq P$ the optimum schedule length is bounded by $O\left(\max_{v \in \Gamma} A_P(\Gamma[v^+], v)\right)$ from above and by $\Omega\left(\max_{v \in \Gamma} A_P(\Gamma[v^-], v)\right)$ from below, however the lower bound is not tight for any power scheme P with $L_{+\epsilon} \preceq P$ (where $\epsilon > 0$) and can be as far from the optima as by a factor of $\Omega(n)$.

Finding constant factor approximation algorithms for nontrivial non-linear power schemes remains an open problem.

Acknowledgment. The author greatly appreciates the comments and suggestions of the anonymous reviewers of the paper.

References

1. Chafekar, D., Kumar, V.S.A., Marathe, M.V., Parthasarathy, S., Srinivasan, A.: Approximation Algorithms for Computing Capacity of Wireless Networks with SINR Constraints. In: Proc. of 27th Annual IEEE Conference on Computer Communications, INFOCOM (2008)
2. Chafekar, D., Kumar, V.S.A., Marathe, M.V., Parthasarathy, S., Srinivasan, A.: Cross-Layer Latency Minimization in Wireless Networks with SINR Constraints. In: Proc. of 26th Annual IEEE Conference on Computer Communications, INFOCOM (2007)
3. Fanghänel, A., Keßelheim, T., Vöcking, B.: Improved Algorithms for Latency Minimization in Wireless Networks. In: Albers, S., Marchetti-Spaccamela, A., Matias, Y., Nikoletseas, S., Thomas, W. (eds.) ICALP 2009, Part II. LNCS, vol. 5556, pp. 447–458. Springer, Heidelberg (2009)
4. Goussevskaia, O., Halldórsson, M.M., Wattenhofer, R., Welzl, E.: Capacity of Arbitrary Wireless Networks. In: 28th Annual IEEE Conference on Computer Communications, INFOCOM (2009)

5. Goussevskaia, O., Oswald, Y., Wattenhofer, R.: Complexity in Geometric SINR. In: ACM International Symposium on Mobile Ad Hoc Networking and Computing, MOBIHOC (2007)
6. Gronkvist, J., Hansson, A.: Comparison between Graph-Based and Interference-Based STDMA scheduling. In: Proc. of ACM International Symposium on Mobile Ad Hoc Networking and Computing, MobiHoc (2001)
7. Halldórsson, M.M.: Wireless scheduling with power control (September 2010), http://arxiv.org/abs/1010.3427
8. Halldórsson, M.M., Mitra, P.: Nearly Optimal Bounds for Distributed Wireless Scheduling in the SINR Model. In: Aceto, L., Henzinger, M., Sgall, J. (eds.) ICALP 2011, Part II. LNCS, vol. 6756, pp. 625–636. Springer, Heidelberg (2011)
9. Halldórsson, M.M., Mitra, P.: Wireless Capacity with Oblivious Power in General Metrics. In: Proc. of ACM-SIAM Symposiun on Discrete Algorithms, SODA (2011)
10. Halldórsson, M.M., Wattenhofer, R.: Wireless Communication is in APX. In: Albers, S., Marchetti-Spaccamela, A., Matias, Y., Nikoletseas, S., Thomas, W. (eds.) ICALP 2009, Part I. LNCS, vol. 5555, pp. 525–536. Springer, Heidelberg (2009)
11. Hardy, G.H., Littlewood, J.E., Pólya, G.: Inequalities. Cambridge University Press (1934)
12. Keßelheim, T.: A Constant-Factor Approximation for Wireless Capacity Maximization with Power Control in the SINR Model. In: Proc. of 22nd ACM-SIAM Symposium on Discrete Algorithms, SODA (2011)
13. Kesselheim, T., Vöcking, B.: Distributed Contention Resolution in Wireless Networks. In: Lynch, N.A., Shvartsman, A.A. (eds.) DISC 2010. LNCS, vol. 6343, pp. 163–178. Springer, Heidelberg (2010)
14. Moscibroda, T., Wattenhofer, R.: The Complexity of Connectivity in Wireless Networks. In: Proc. of 25th Annual IEEE Conference on Computer Communications, INFOCOM (2006)
15. Moscibroda, T., Wattenhofer, R.: How Optimal are Wireless Scheduling Protocols? In: Proc. of 26th Annual IEEE Conference on Computer Communications, INFOCOM (2007)
16. Moscibroda, T., Wattenhofer, R., Weber, Y.: Protocol Design Beyond Graph-Based Models. Hot Topics in Networks, HotNets (2006)
17. Rappaport, T.: Wireless Communications: Principles and Practice. Prentice Hall (2002)
18. Tonoyan, T.: On the Capacity of Oblivious Powers. In: Erlebach, T., Nikoletseas, S., Orponen, P. (eds.) ALGOSENSORS 2011. LNCS, vol. 7111, pp. 225–237. Springer, Heidelberg (2012)
19. Tonoyan, T.: On the Problem of Wireless Scheduling with Linear Power Levels, http://arxiv.org/abs/1107.4981

Polynomial Time Approximation Algorithms for Localization Based on Unknown Signals

Johannes Wendeberg and Christian Schindelhauer

Department of Computer Science, University of Freiburg, Germany
{wendeber,schindel}@informatik.uni-freiburg.de

Abstract. We consider the problem of anchor-free self-calibration of receiver locations using only the reception time of signals produced at unknown locations and time points. In our settings the receivers are synchronized, so the time differences of arrival (TDOA) of the signals arriving at the receivers can be calculated. Given the set of distinguishable time points for all receivers the task is to determine the positions of the receivers as well as the signal sources.

We present the first polynomial time approximation algorithms for the minimum problem in the plane, in which the number of receivers is four, respectively the number of signals is three. For this, we first consider the problem that the time points of m signals are jittered by at most some $\epsilon > 0$. We provide an algorithm which tests whether n given receiver positions are feasible with respect to m unknown sender positions with a run-time of $\mathcal{O}(nm^2)$ and we provide an algorithm with run-time $\mathcal{O}(nm \log m)$ which tests the feasibility of m given sender positions for n unknown sender positions. Using these tests, we can compute all possible receiver and signal source positions in time $\mathcal{O}\left(\left(\sqrt{2}/\epsilon\right)^{2n-3} n^2 m\right)$, respectively $\mathcal{O}\left(\left(\sqrt{2}/\epsilon\right)^{2m-3} nm \log m\right)$.

1 Introduction

The problem of location awareness of computing devices plays an important role in many engineering fields. A popular technique for acquiring the positions of devices is hyperbolic multilateration, as for example used in the LORAN and DECCA systems, global navigation satellite systems (GNSS), or cellular phone tracking in GSM networks.

In hyperbolic localization the position of a signal origin is located by a set of synchronized receivers. The times of arrival or time differences of arrival (TDOA) of a signal determine a system of hyperbolic equations. The benefit of this TDOA multilateration is that no control over the signal source is needed. Arbitrary sources of sound can be overheard, or the signal of radio emitters. Dedicated signal sources, as for example a tracked moving ultrasound beacon, can be primitive and cheap. It needs only to emit ultrasound pulses – no control signal is required.

In an extension the positions of anchors are *not* given a priori and are obtained as a result of the calculations. We present an approach to solve the problem in

A. Bar-Noy and M.M. Halldórsson (Eds.): ALGOSENSORS 2012, LNCS 7718, pp. 132–143, 2013.

an approximation scheme yielding the best explanation of receiver locations for given time differences of arrival, up to a threshold of ϵ, which is chosen on the basis of the input error.

1.1 Related Work

The problem of TDOA localization has been widely researched and is present in literature. In conventional hyperbolic multilateration with fixed receiver positions the problem of signal localization is solved in closed form [4], [15], or by iterative approaches, and position estimates can be refined using Kalman filtering [5].

TDOA times are calculated by discrete timestamping [1], [6] or by cross correlation of audio streams [21]. Sources of information may be ultrasound signals [22], often in combination with RF signals. These are solved as *Time of Arrival* approaches [13], [16].

The problem of self-localization of receivers using only TDOA data and no anchor points is seldom considered in literature. In [11] a combination with DOA ("direction of arrival") data is presented as an initialization aid. In some cases the signals are assumed to originate from a large distance, the "far-field case" [6], [7], [20]. For near-field signals in the unit disc an upper bound of the error is shown in [17]. In [14] an approach is presented under the assumption that speakers are close to the receivers. In some contributions iterative algorithms are used to compute a solution [1], [22]. However, they tend to get stuck in local minima of the error function.

If many receivers are available there exists a closed form solution. For at least eight receivers in the plane, respectively ten receivers in three-dimensional space, the problem can be solved using matrix factorization [12].

Our refinement approach is inspired by branch-and-bound algorithms [9] usually applied to discrete problems. An algorithm for integer programming was first presented in [3]. By extension with Linked Ordered Sets it can be applied to nonlinear integer problems [8]. We saw some contributions suggesting the application of branch-and-bound to nonlinear nonconvex functions for global optimization [10]. However, problem solvers using branch-and-bound algorithms for multi-dimensional continuous space are rare.

2 Problem Setting

Let M_i $(1 \leq i \leq n)$ be a set of receivers and S_j $(1 \leq j \leq m)$ a set of signal origins, all positioned in the unit square of the Euclidean plane $\mathbb{R}^2$. Both, the receiver and the signal positions are unknown. The signals are emitted at unknown times u_j. Each signal S_j is propagated to each synchronized receiver M_i in a direct line with constant signal speed c and is detected at times t_{ij}. These times are the sole information given in the system. We normalize $c = 1$ for the simplicity and so, the signals and receivers and the event times satisfy the nm signal propagation equations for all $i \in \{1, \ldots, n\}$, $j \in \{1, \ldots, m\}$

$$t_{ij} - u_j = \|M_i - S_j\| \tag{1}$$

where $\|\cdot\|$ denotes the Euclidean norm. From these equalities we cannot derive absolute coordinates. We remove transitional, rotational and mirroring symmetries by placing the first receiver M_1 at $(0,0)$, placing the second receiver on the x-axis and place the third receiver on the half-plane with positive y-coordinates.

The degrees of freedom $\mathcal{G}_2$ have been considered in [19] and [22]. Each of the nm equations decrements the degree of freedom. For n receivers and m signals we have $\mathcal{G}_2(n,m) = 2n+3m-nm-3$ degrees of freedom in the plane. If $\mathcal{G}_2(n,m) > 0$ then either infinitely many solutions or no solutions exist. Only if $\mathcal{G}_2(n,m) \leq 0$ the solution space can be reduced to a single solution. However, also no solutions, a constant number of solutions or infinitely many solutions may exist in this case, depending on the input. We conjecture that for $\mathcal{G}_2(n,m) < 0$ no side solutions exist if the input is derived from signal sources and receivers in general position, i.e. this conjecture holds with probability 1 for random positions from the unit square.

Definition 1. ***Anchorless localization problem*** *(self-calibration based on TDOA): Given the exact time points t_{ij} compute the positions of senders and receivers such that Equation (1) is satisfied.*

The problem is only solvable, if $n = 4, m \geq 5$, or $n = 5, m \geq 4$, or $n \geq 6, m \geq 3$. If the system is only slightly over-constrained, there is some chance of ambiguity. For example, for $n = 4, m = 5$ the number of solutions is bounded by 344 [19]. For $n = 6, m = 3$ at most 150 solutions exist. We have found some inputs where at least two solutions exist for $n = 4$ and $m = 5$. However, their exact number and probability is not known at the moment and are part of future work.

If the system is highly over-constrained, i.e. $n \geq 8$, $m \geq 4$ yielding $\mathcal{G}_2(n,m) \leq -7$, then Pollefeys et al. [12] showed how this nonlinear equation system can be transformed into a linear equation system, which allows for a polynomial time solution with run-time $\mathcal{O}(m^2n^2)$. For other values only heuristic algorithms are known, which do not generate solutions for all inputs, as in [1] and [22].

Clearly, there are precision limits for the inputs t_{ij}. So, we consider the relaxed inequalities which assumes an error bound $\epsilon > 0$ for all $i \in \{1,\ldots,n\}$, $j \in \{1,\ldots,m\}$.

$$t_{ij} - \epsilon \leq u_j + \|M_i - S_j\| \leq t_{ij} + \epsilon \tag{2}$$

If we can compute a solution which satisfies these inequalities we call this an ϵ-approximation. Note that we are aware that the positions of the receivers or signals might be further than ϵ from the real position. Furthermore, in an ambiguous problem it is possible that we approximate multiple solutions which are not even close to the original positions. Then, we cannot decide which one is the correct solution. However, it is the best one can achieve given the error-prone problem setting.

Definition 2. ***Approximation problem of anchorless localization****: Given the time points t_{ij} and u_j and an error margin ϵ compute a possible set of positions of senders and receivers such that the inequalities (2) are satisfied, if they exist.*

This problem is not addressed by Pollefeys' algorithm. Furthermore, for small numbers of receivers ($n < 8$) or small numbers of signals ($m = 3$) no solution is known. We have presented a solution for large numbers of senders and receivers randomly distributed in a unit disk [17]. For synchronized receivers we can compute in time $\mathcal{O}(nm)$ an approximation of the correct relative positions within an absolute error margin of $\mathcal{O}\left(\frac{\log^2 m}{m^2}\right)$ with probability $1 - m^{-c} - e^{-c'n}$.

For small n and m there is a naïve solution for finding receiver and signal positions. For this one can test all $(2^{-\frac{3}{2}}\epsilon)^{3-2n-2m}$ positions of senders and receivers in a grid of cell size $\frac{1}{2\sqrt{2}}\epsilon$ whether they satisfy the inequalities (2). From the positions the signal time can be easily computed and the inequalities can be checked in time $\mathcal{O}(nm)$. When this exhaustive search has tested receivers and signals within the distance $\frac{1}{2}\epsilon$ from the correct solution this implies an overall error of ϵ, if there exists a solution.

Theorem 1. *The naïve approximation algorithm solves the approximation problem in time* $\mathcal{O}((2^{-\frac{3}{2}}\epsilon)^{3-2n-2m}nm)$.

Proof. Follows by testing all positions of the $2n+2m-3$ dimensional grid. Note that the constant three follows since we set M_1 to the origin and M_2 to the x-axis to reduce symmetries. The production time u_j of the signal can be computed from the distances, therefore we need to search only space and not time.

3 Polynomial Time Approximation for Small Numbers of Receivers

This approximation algorithm consists of two components. The first component tests for a given set of receivers if there exists a set of signal sources satisfying the constraints. The second component is a recursive tree search to find the receiver positions by applying the first test with increasing precision.

3.1 A Test for the Feasibility of Receiver Positions

If the receiver positions are known the location of the signal sources can be computed very efficiently. This observation inspires the following test algorithm. The problem of given receiver positions is to find signal sources which satisfy the inequalities (2).

We combine two receiver times t_{kj} and $t_{\ell j}$ for one signal source j to the time difference of arrival $\Delta t_{k\ell j} = t_{kj} - t_{\ell j}$ and yield the hyperbolic equation

$$\Delta t_{k\ell j} = \|M_k - S_j\| - \|M_\ell - S_j\| \tag{3}$$

for the exact solution which describes a hyperbola, called $\mathcal{H}(k, \ell, j, \Delta t_{k\ell j})$ with M_k and M_ℓ as focal points and S_j residing on the curve. For the approximation problem we consider the inequality

$$|\Delta t_{k\ell j} - (\|M_k - S_j\| - \|M_\ell - S_j\|)| \leq \epsilon \tag{4}$$

where ϵ is an upper bound of the error.

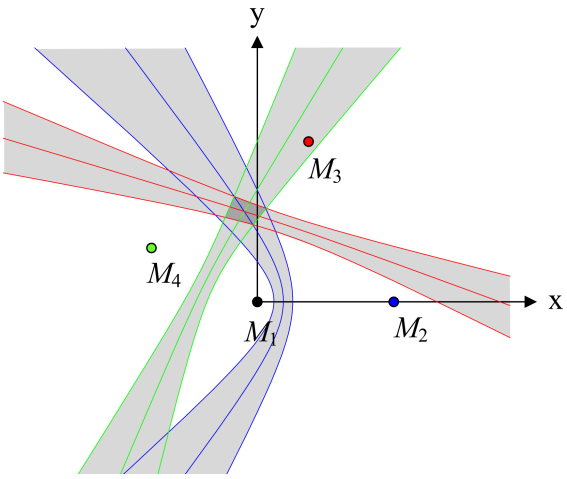


Fig. 1. The measured time difference $\Delta t_{k\ell j}$ between M_k and M_ℓ yields a hyperbola (solid lines). Inequation (4) bounds a region of uncertainty where S_j can reside (grey regions), here depicted for the origin M_1 and three other receivers.

Lemma 1. *The inequality (4) for $k = 1$, $\ell \in \{1, \ldots, n\}$ and $j \in \{1, \ldots, m\}$ follows from the $\frac{1}{2}\epsilon$-approximation problem of localization.*

From inequality (4) for $k = 1$, $\ell \in \{1, \ldots, n\}$ and $j \in \{1, \ldots, m\}$ follows a solution for the ϵ-approximation problem.

Proof.

"⇒": Using the inequalities (2) and eliminating u_j leads to inequality (4) with 2ϵ.
"⇐": Choose $u_j = t_{1j} - \|M_1 - S_j\|$. Then, the approximation inequality (2) follows for $k > 1$. Note that for $k = 1$ we obtain Equation 1.

The inequality (4) describes an *uncertainty band* for the position of the signal source M_j enclosed by the hyperbolas $\mathcal{H}(1, \ell, j, \Delta t_{1\ell j} - \epsilon)$ and $\mathcal{H}(1, \ell, j, \Delta t_{1\ell j} + \epsilon)$, see Fig. 1. We compute the intersections of these areas for all M_ℓ for $\ell > 1$. If the intersections of these areas are non-empty, then for these receivers positions a signal source exists. If these areas exist for all signal sources, then there is a solution to the approximation problem. So, the test problem is reduced to a computational geometry problem.

Lemma 2. *The intersection of the $n - 1$ uncertainty bands for a signal source S_j can be computed in time $\mathcal{O}(n^2)$.*

Proof. We use a sweep line technique where the sweep line is a half-line starting at the origin M_1 rotating around it. This is motivated by the observation that such a sweep line intersects each hyperbola $\mathcal{H}(1, \ell, j, \Delta t_{1\ell j} \pm \epsilon)$ at most once. Furthermore in polar coordinates the intersection band is beyond the union of all hyperbolas $\mathcal{H}(1, \ell, j, \Delta t_{1\ell j} - \epsilon)$ and below all hyperbolas $\mathcal{H}(1, \ell, j, \Delta t_{1\ell j} + \epsilon)$.

In this proof we compute the curve in polar coordinates which is defined by the farthest points of all hyperbolas $\mathcal{H}(1, \ell, j, \Delta t_{1\ell j} - \epsilon)$ on the sweep-line with angle α. Then we compute the curve of the closest points of all hyperbolas $\mathcal{H}(1, \ell, j, \Delta t_{1\ell j} + \epsilon)$ on the sweep-line with angle α. We describe both points by the section-wise definition of the curves. Then, we test in linear time (depending on the number of sections) whether between the $-\epsilon$-curve and the ϵ-curve an area exists.

Curve of farthest points of all hyperbolas $\mathcal{H}(1, \ell, j, \Delta t_{1\ell j} - \epsilon)$. Each hyperbola is only defined in a sector framed by the central angle interval. So, first we compute the intersection of the angle intervals, which can be done by sorting start and end angles in time $\mathcal{O}(n \log n)$. Note that two hyperbolas intersect in at most two points and that these points can be computed directly. By the definition of the Davenport-Schinzel-sequences [18] the number of intersections in the curve is bounded by $\lambda_2(n-1) = 2n-3$.

We compute the curve by starting with the hyperbola $\mathcal{H}(1, i_0, j, \Delta t_{1 i_0 j} - \epsilon)$ corresponding to the start of the intersected angle interval. Then we compute for this hyperbola all intersections with other hyperbolas $\mathcal{H}(1, \ell, j, \Delta t_{1\ell j} - \epsilon)$ and test whether they cross this hyperbola. We take the first (right-handed seen from the origin) such hyperbola following the sweep-line approach. So, we get the next hyperbola corresponding to receiver M_{j_1} in time $\mathcal{O}(n)$. We repeat this search getting receivers $M_{j_2}, M_{j_3}, \ldots$ until we have arrived at the end of the intersecting angle interval. Since the number of intersection is bounded by $2n-3$, the overall run-time is $\mathcal{O}(n^2)$.

Curve of closest points of all hyperbolas $\mathcal{H}(1, \ell, j, \Delta t_{1\ell j} + \epsilon)$. Now we have to compute the union of all angle intervals of the hyperbolas $\mathcal{H}(1, \ell, j, \Delta t_{1\ell j} + \epsilon)$. Again this can be done by sorting the angles in time $\mathcal{O}(n \log n)$. If the curve is not defined over the full range we can use the above method for each disjoint interval. In the other case we need to compute a starting point. For this we compute the nearest points (seen from the origin M_1) for each of the $n-1$ hyperbolas which can be done in closed form. The hyperbola segment around this point is part of the solution. Starting from this hyperbola we can use the analogous algorithm to explore the segments of the curve. Note that in this case the number of intersections is slightly higher, since one additional intersection may occur because of the circular nature of the functions yielding $\lambda_2(n-1) + 1 = 2n-2$ hyperbola segments. Again we needs time $\mathcal{O}(n)$ for finding the next segment. So, the overall computation time is again $\mathcal{O}(n^2)$.

Testing the non-emptiness of the intersection. Both curves are given as sorted lists of the hyperbolas, according to the angles. Each of them has at most $2n-1$ segments. We join the set of angles and test for each interval on which the hyperbolas lie and whether intersections occur. For each of the intervals this can be done in constant time, resulting in an additional effort of time $\mathcal{O}(n)$.

Lemma 3. *The test of the feasibility of receiver positions can be performed in time* $\mathcal{O}(n^2 m)$.

Proof. This follows from repeating the above test for all m signals.

3.2 Recursive Search for the Receiver Positions

Now, one could enumerate all grid positions of distance $\frac{1}{\sqrt{2}}\epsilon$ and use the test to solve the approximation problem resulting in run-time $\mathcal{O}((\sqrt{2}/\epsilon)^{2n-3} m n^2)$ much alike the trivial algorithm. However, there is a much more efficient method which reduces the time behavior in practical tests. In fact empirical tests point towards an average run-time of $O((-\ln \epsilon) n^2 m)$ for the now following approach.

We consider a recursive tree construction shown in Algorithm 1, where the $2n-3$-dimensional subgrid is repartitioned by a factor of two in each iteration. It uses the feasibility test described in the previous subsection. In the uppermost grid (consisting of only one cell) we choose $\hat{\epsilon} = \sqrt{2}$ and decrease this value in each level by a factor of two. If we have reached $\hat{\epsilon} \leq \epsilon_{\text{target}}$ the search algorithm stops. For each level we discard non-feasible sub-cells. This way, we avoid exhaustive search by pruning the search tree at a higher level. Simulations indicate that in the long run the number of sub-cells remains at most in the hundreds. However, we can show only the run-time of the trivial method which is not very efficient.

Theorem 2. *Algorithm 1 solves the approximation problem in time* $\mathcal{O}((\sqrt{2}/\epsilon)^{2n-3} m n^2)$.

Proof. The theorem follows already from the trivial test algorithm which tests all points in the grid. Note that the improved Algorithm 1 also satisfies this bound.

We prove the correctness of Algorithm 1. Clearly, if the algorithm finds a solution then Lemma 1 implies that it is a solution for the approximation problem. It remains to show that a solution is always found.

Now consider the solution of the localization problem of $\hat{M}_i$. Then, in a grid of cell size s for each receiver $\hat{M}_i$ there is a point M_i within distance $\sqrt{2}s$ with the exception of $\hat{M}_1$ which we define as $M_1 = (0, 0)$. Inequality (4) holds for $\epsilon = \sqrt{2}s$. Therefore this set of receivers would not be discarded in the feasibility test as long as $\epsilon \geq \sqrt{2}s$. Algorithm 1 ensures that sub-cells are only erased from the queue until ϵ is smaller. Therefore, the cell with center M_i is not removed from the queue unless a smaller cell containing the solution is inserted.

4 Polynomial Time Approximation for Small Numbers of Signals

Now, we consider the case where the number of signal sources is small, e.g. $m = 3$. Now, we search possible signal locations and test whether these positions are feasible.

Algorithm 1. Breadth-first search for ϵ-environment for n receivers

Require: Given initial guess of receiver positions $M := \{M_1, \ldots, M_n\}$, receiver times $(t_{i,j})_{i\in[n],j\in[m]}$, target ϵ_{target}

queue $Q \leftarrow \emptyset$
$M \leftarrow (0,0)^n$,
$Q \leftarrow Q.\text{enqueue}\,((0, M))$
repeat
  $(d, M') \leftarrow Q.\text{pop}()$
  for all $b_1, \ldots, b_{2n-3} \in \{-1, 1\}^{2n-3}$ **do**
    $\tilde{M} = M' + 2^{-d-1} \cdot ((0,0), (b_1, 0), (b_2, b_3), \ldots, (b_{2n-2}, b_{2n-3}))$
    if ReceiverPositionIsFeasible$\left(\tilde{M}, (t_{*,*}), 2^{-d-\frac{1}{2}}\right)$ **then**
      $Q \leftarrow Q.\text{enqueue}\left((d+1, \tilde{M})\right)$
    end if
  end for
until $Q = \emptyset$ **or** $2^{-d-\frac{1}{2}} \leq \epsilon_{\text{target}}$
return $Q.\text{pop}()$

4.1 A Test for the Feasibility of Signal Source Positions

We revisit the inequalities (4), but now we consider M_ℓ as a variable while $M_k = M_1$ and S_j is fixed. Given the positions $S_1, \ldots, S_m$, we consider all possible locations for a receiver M_i. These are intersections of disks and complements of disks. If the intersections are non-empty, then the signal locations are feasible.

Lemma 4. *The intersection of the m disks and m disk complements can be computed in time $O(m \log m)$.*

Proof. The intersection of m disks can be computed in time $\mathcal{O}(m \log m)$ [2]. The union of m disks can be also computed in time $\mathcal{O}(m \log m)$ [2], which describes the intersection of the disk complements. Using a sweep line technique one can now test whether the intersection of these two areas is empty. The time for this is also bound by $\mathcal{O}(m \log m)$.

Given the boundaries of the intersections a sweep line method can compute the intersecting area. The sweep line events occur when the sweep line hits a segment of the two boundary paths the first time, when segments intersect, or when the sweep line leaves a segment. All these events can be computed in time $\mathcal{O}(m \log m)$ in advance or (for the intersection of the two boundaries) in time $\mathcal{O}(\log m)$ when a new boundary segment is introduced. Applying the sweep line method computes the wanted intersection in time $\mathcal{O}(m \log m)$, since at most $\mathcal{O}(m)$ events take place.

Lemma 5. *The test for the feasibility of signal positions can be performed in time $O(nm \log m)$.*

Proof. This follows from repeating the test for all $n - 1$ receivers.

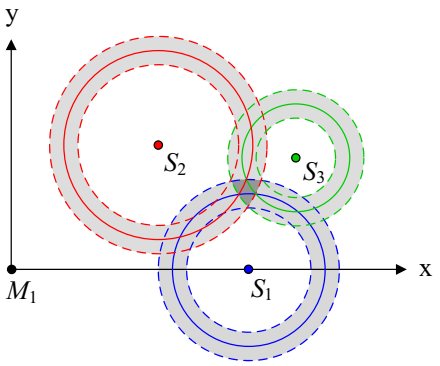


Fig. 2. Given a time difference Δt between a fixed receiver M_1 and an unknown receiver M_ℓ. Now, a fixed location for S_1, S_2, and S_3 is assumed. The hyperbolic inequality (4) is satisfied if the unknown receiver M_ℓ resides in a circular epsilon environment (grey regions).

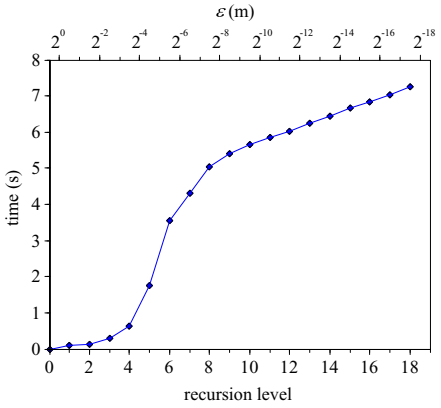


Fig. 3. Total run-time of an example with four receivers and 20 signals on an Intel Core-i5 quad-core CPU to achieve precision ϵ. The recursion level is $\delta = 0.5 - \log_2(\epsilon)$. The run-time is proportional to the number of processed nodes.

4.2 Recursive Search for the Signal Positions

This search is completely analogous to Section 3.2 and Algorithm 1 while we now search for signal positions.

Theorem 3. *The analogous search approximation algorithm solves the problem in time* $\mathcal{O}\left(\left(\sqrt{2}/\epsilon\right)^{2m-3} nm \log m\right)$.

Proof. The trivial grid oriented algorithm testing all positions of in a grid of cell size $\epsilon/\sqrt{2}$ combined with the feasibility test above yields the run-time.

Again a tree-based search performs better for real-world inputs, but does not give better worst-case bounds.

5 Empirical Results

We concentrate on the case of four receivers and a large number of signal sources, since the practical impact of this problem is higher. We have implemented our algorithm in a computer algebra system as well as in C++. As the inner nodes of the search tree are independent the problem is inherently parallel. We profit from this characteristic and execute multiple threads on modern multi-core CPUs.

For our experiments we generate the positions of four receivers randomly in the unit circle. They are transformed such that the first location is in the origin, the second is on the positive leg of the x-axis, and the third receiver is in the upper two quadrants.

Now m signals are sampled at random positions in the unit circle. For m we choose a series from 5 to 50 signals with 100 runs for each number of signals. The time differences of arrival are calculated and passed to the algorithm, which is the only information given to the algorithm.

We have evaluated the algorithm in terms of inspected search nodes, search queue length, duration and correctness of the calculation, i.e. the receiver positions.

In some under-determined cases with few signal sources, or malicious receiver positions, where any two receivers are very close to each other, we observe runtimes as indicted by the worst case analysis. Then, the width of the search tree grows rapidly resulting in high run-times. In these cases we abort the algorithm after 10 minutes and mark the attempt failed (although the algorithm would eventually find the solution).

The number of required evaluation steps depends on the location characteristics of the receivers, on the traversal strategy through the tree, and on the number of signals. In our simulations we choose breadth-first search, which is the slowest search type, but with deterministic characteristics. Then, given a sufficient number of signals, the number of traversed nodes varies between 10^4 and 10^7 with a cumulation at 10^5. With decreasing number of signals the algorithm has increased difficulty to eliminate possible locations, increasing the number of steps by a factor of 10^1 and more.

On an Intel Core-i5 machine we could process a number of 10^5 nodes in about $4-8$ seconds, running on four processor cores (Fig. 3). A typical execution time given 40 signals is 8 seconds, which is the mode of the distribution of runtimes, with some ill-conditioned settings raising the mean to 10.5 seconds. Alltogether, the typical execution time is 10 seconds with a mean of 44 seconds.

At the current stage our implementation is too slow for real-time applications on standard PC hardware. However the prospects are great, as the search algorithm can be so easily run in parallel. On typical computers, and even smartphones, the number of processing cores increases permanently and our algorithm can benefit with efficiency: Our search algorithm does speed up with both higher core performance and higher number of cores, other than iterative algorithms which usually profit only from increases in per-core performance.

6 Conclusions and Future Work

In this contribution we have presented what is, to our knowledge, the first solution for the TDOA-Self-Localization problem given the minimum number of four receivers. In our model the uncertainties of TDOA measures are expressed by an ϵ-approximation scheme, returning the best explanation of receiver locations for the given time-of-arrival data of signals from unknown locations. If the data is precise up to an error of ϵ then also the receiver locations are determined up to an error in the order of ϵ.

For refining the estimation of our ϵ-model we use a fully polynomial time approximation scheme running in time $\mathcal{O}((\sqrt{2}/\epsilon)^{2n-3}n^2m)$ for the receiver problem

and in $\mathcal{O}((\sqrt{2}/\epsilon)^{2m-3}nm \log m)$ for the analogous problem of estimating small numbers of at least three signals. This implies the following corollary.

Corollary 1. *The approximation of the four receiver localization problem can be solved in time $\mathcal{O}(\epsilon^{-5}m)$ for m signal sources. The problem of the three signal localization can be solved in time $\mathcal{O}(\epsilon^{-5}n)$ for n receivers.*

We have implemented the algorithm in a multi-threaded simulation of randomized receiver locations in the unit disc. We could show the feasibility of our approach and we could show that we traverse the search tree in a couple of seconds in most cases.

In some cases our approach suffers from an ill-conditioned configuration of the receiver locations, i.e. some of the four receivers are close to each other or near to a line, rendering the problem close to under-determined. Then, our algorithm is forced to generate a very large search tree, resulting in a long duration for the traversal. However, when we find a solution we are guaranteed that it is correct, up to the order of ϵ.

One open problem is whether the intersection of n halfspaces bordered by hyperbolas can be computed in time $\mathcal{O}(n \log n)$ like the intersection of disks. To our knowledge nobody has addressed this non-trivial problem so far.

We have seen that our algorithm suffers from a large search tree in some cases, and according to that long execution times. There are worst-case inputs where this is inevitable, i.e. if the receivers are located within a radius of ϵ. But also for non-degenerated inputs we see plenty of room for improvements, as we use Breadth-First-Search for now to traverse the tree in a deterministic manner, which allows to draw conclusion about the expected run-time.

If we use Depth-First-Search we can use heuristics to choose branches of the search tree. The task is to develop such a model using elementary approximation schemes as for example presented in [6] and [17].

Acknowledgements. This work has partly been supported by the German Research Foundation (Deutsche Forschungsgemeinschaft, DFG) within the Research Training Group 1103 (Embedded Microsystems). We thank Wolfram Burgard for the discussions that inspired us to come up with this solution.

References

1. Biswas, R., Thrun, S.: A Passive Approach to Sensor Network Localization. In: Proceedings of the 2004 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2004), vol. 2, pp. 1544–1549 (2004)
2. Brown, K.Q.: Geometric Transforms for Fast Geometric Algorithms. Ph.D. thesis, Carnegie-Mellon University Pittsburgh PA, Dept. of Computer Science (1979)
3. Dakin, R.J.: A tree-search algorithm for mixed integer programming problems. The Computer Journal 8(3), 250–255 (1965)
4. Gillette, M.D., Silverman, H.F.: A Linear Closed-Form Algorithm for Source Localization From Time-Differences of Arrival. IEEE Signal Processing Letters 15, 1–4 (2008)

5. Hongyang, C., Ping, D., Yongjun, X., Xiaowei, L.: A Robust Location Algorithm with Biased Extended Kalman Filtering of TDOA Data for Wireless Sensor Networks. In: Proceedings of the 2005 International Conference on Wireless Communications, Networking and Mobile Computing, vol. 2, pp. 883–886. IEEE (2005)
6. Janson, T., Schindelhauer, C., Wendeberg, J.: Self-Localization Application for iPhone using only Ambient Sound Signals. In: Proceedings of the 2010 International Conference on Indoor Positioning and Indoor Navigation (IPIN), pp. 259–268 (November 2010)
7. Janson, T., Schindelhauer, C., Wendeberg, J.: Self-localization Based on Ambient Signals. In: Scheideler, C. (ed.) ALGOSENSORS 2010. LNCS, vol. 6451, pp. 176–188. Springer, Heidelberg (2010)
8. Kennedy, D.: Some branch and bound techniques for nonlinear optimization. Mathematical Programming 42(1), 147–157 (1988)
9. Land, A.H., Doig, A.G.: An Automatic Method of Solving Discrete Programming Problems. Econometrica 28(3), 497–520 (1960)
10. Lawler, E.L., Wood, D.E.: Branch-And-Bound Methods: A Survey. Operations Research 14(4), 699–719 (1966)
11. Moses, R.L., Krishnamurthy, D., Patterson, R.M.: A Self-Localization Method for Wireless Sensor Networks. EURASIP Journal on Advances in Signal Processing, 348–358 (2003)
12. Pollefeys, M., Nister, D.: Direct computation of sound and microphone locations from time-difference-of-arrival data. In: IEEE International Conference on Acoustics, Speech and Signal Processing, pp. 2445–2448. IEEE (2008)
13. Priyantha, N.B., Chakraborty, A., Balakrishnan, H.: The Cricket Location-Support System. In: MobiCom 2000: Proceedings of the 6th Annual International Conference on Mobile Computing and Networking, pp. 32–43 (2000)
14. Raykar, V., Duraiswami, R.: Automatic position calibration of multiple microphones. In: Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP 2004), vol. 4, pp. iv–69 IEEE (2004)
15. Rui, Y., Florencio, D.: New direct approaches to robust sound source localization. In: Proceedings of IEEE ICME 2003, pp. 6–9. IEEE (2003)
16. Savvides, A., Han, C.C., Strivastava, M.B.: Dynamic Fine-Grained Localization in Ad-Hoc Networks of Sensors. In: Proceedings of the 7th Annual International Conference on Mobile Computing and Networking, pp. 166–179. ACM (2001)
17. Schindelhauer, C., Lotker, Z., Wendeberg, J.: Network Synchronization and Localization Based on Stolen Signals. In: Kosowski, A., Yamashita, M. (eds.) SIROCCO 2011. LNCS, vol. 6796, pp. 294–305. Springer, Heidelberg (2011)
18. Sharir, M., Agarwal, P.K.: Davenport Schinzel sequences and their geometric applications. Cambridge Univ. Press, New York (1995)
19. Stewénius, H.: Gröbner Basis Methods for Minimal Problems in Computer Vision. Ph.D. thesis, Lund University (April 2005)
20. Thrun, S.: Affine Structure From Sound. In: Proceedings of Conference on Neural Information Processing Systems (NIPS). MIT Press, Cambridge (2005)
21. Valin, J.M., Michaud, F., Rouat, J., Létourneau, D.: Robust Sound Source Localization Using a Microphone Array on a Mobile Robot. In: Proceedings of the International Conference on Intelligent Robots and Systems (IROS), pp. 1228–1233 (2003)
22. Wendeberg, J., Höflinger, F., Schindelhauer, C., Reindl, L.: Anchor-free TDOA Self-Localization. In: Proceedings of 2011 International Conference on Indoor Positioning and Indoor Navigation, IPIN (2011)

Author Index

Albagli-Kim, Sivan 6
Aschner, Rom 18

Belke, Lukas 30

Chan, David Yu Cheng 42
Chun, Jinhee 54

Daltrophe, Hadassa 66
DiPippo, Lisa C. 93
Disser, Yann 78
Dolev, Shlomi 66

Epstein, Leah 6

Ghosh, Subir Kumar 78

Hoffmann, Stefan 90

Joseph, Shaun N. 93

Katz, Matthew J. 18
Kesselheim, Thomas 1, 30
Kirkpatrick, David 42
Koster, Arie M.C.A. 30

Leone, Pierre 105
Liang, Hongyu 108
Lotker, Zvi 66
Lou, Tiancheng 108

Mihalák, Matúš 78
Morgenstern, Gila 18

Schiller, Elad Michael 105
Schindelhauer, Christian 132
Shachnai, Hadas 6
Shioura, Akiyoshi 54
Suri, Subhash 4

Tamir, Tami 6
Tan, Haisheng 108
Tien, Truong Minh 54
Tokuyama, Takeshi 54
Tonoyan, Tigran 120

Vöcking, Berthold 30

Wang, Amy Yuexuan 108
Wanke, Egon 90
Wendeberg, Johannes 132
Widmayer, Peter 78

Yu, Dongxiao 108