DOI:10.15774/PPKE.ITK.2019.006

Applying neural network based algorithms in communication technology

A dissertation submitted for the degree of

Doctor of Philosophy (Ph.D.)

by

Dávid Tisza

Scientific advisors:

Dr. János Levendovszky

dr. András Oláh

Pázmány Péter Catholic University

Faculty of Information Technology and Bionics

Roska Tamás Doctoral School of Sciences and Technology

Budapest, 2018

DOI:10.15774/PPKE.ITK.2019.006

Agnus Dei, qui tollis peccáta mundi, miserére nobis.
Agnus Dei, qui tollis peccáta mundi, miserére nobis.
Agnus Dei, qui tollis peccáta mundi, dona nobis pacem.

[29]*altera die videt Iohannes Iesum venientem ad se et ait ecce agnus Dei qui tollit peccatum mundi* [30]*hic est de quo dixi post me venit vir qui ante me factus est quia prior me erat* [31]*et ego nesciebam eum sed ut manifestaretur Israhel propterea veni ego in aqua baptizans (Jn 1,29-31)*

[29]*Másnap, amikor János látta, hogy Jézus feléje tart, így szólt: „Nézzétek, az Isten Báránya! Ő veszi el a világ bűneit.* [30]*Róla mondtam: A nyomomba lép valaki, aki nagyobb nálam, mert előbb volt, mint én.* [31]*Én sem ismertem, de azért jöttem vízzel keresztelni, hogy megismertessem Izraellel" (Jn 1,29-31)*

[29]*The next day, he saw Jesus coming towards him and said, 'Look, there is the lamb of God that takes away the sin of the world.* [30]*It was of him that I said, "Behind me comes one who has passed ahead of me because he existed before me."* [31]*I did not know him myself, and yet my purpose in coming to baptise with water was so that he might be revealed to Israel.' (Jn 1,29-31)*

DOI:10.15774/PPKE.ITK.2019.006

Acknowledgement

First I would like to give thanks to my supervisor *Dr. János Levendovszky*, to †*prof. Tamás Roska*, to *Dr. Árpád Csurgay*, to *Judit Nyékyné dr. Gaizler*, to *Dr. Péter Szolgay* and to all the present and past directors of the doctoral school, to *Martin Haenggi* and *prof. Géza Kolumbán* for their encouragement, advises and critics of high standards which all modulated the trajectory I traveled so far.

I would like to express my special gratitude to *dr. András Oláh* and to *Péter Vizi*, *dr. Kornél Németh*, *János Pásztor*, *József Somogyi*, *Dániel Süttő* and *Dávid Kauker* for their support, endurance and help, without which this work would not have been completed.

I would like to thank my fellow doctoral school members and co-workers for their community and support, inter alia *András Bojárszky*, *dr. Gergely Treplán*, *dr. Kálmán Tornai*, *dr. Csaba Józsa*, *dr. Reguly István*, *Miklós Koller*, *Imre Juhász*,
Gábor Abonyi, *Péter Pál Porázik*, *Tamás Kosztolánczi*, *Attila Fehér*, *Marcell Tibély*, *Máté Nagy*, *Ferenc Varjasi*, *Alexander Dvorzsák*, *Milán Györki*.

I would like to give thanks to *Katinka Vida Tivadarné* for her precise and attentive support throughout my years in the doctoral school and beyond.

To the members of the finance and registrars department, the members of the deans office, the members of the IT department and for all fellow members of the faculty for their work, which are often not as visible but equally important.

I would also like to give thanks to the members and founders of the *SP CEE Scholarship* program for their friendship, guidance, patience and financial support.The project was supported by the European Union, co-financed by the European Social Fund (EFOP-3.6.3-VEKOP-16-2017-00002).

And I am deeply grateful to my parents, brothers and to the other members of my family: *Kálmán Tisza*, †*Kálmánné Tisza* born *Katalin Adorján*, *Kálmán* and *Levente*, †*Gergelyné Busa Ildikó*, also to my magisters: †*János Gutai*, †*Mária Kovaliczky* and teachers: namely *Katalin Szabó Kálmánné* and *Erika Szeitzné Viski* for building the foundations on which I could set a firm stand.

DOI:10.15774/PPKE.ITK.2019.006

Contents

Acronyms vi

Kivonat viii

Abstract ix

1 Introduction 1
1.1 Applied methodology . . . 2
1.2 Numerical analysis framework . . . 3
1.3 Structure of the dissertation . . . 4

2 Thesis group I - routing with incomplete information in unicast and multicast scenarios 5
2.1 Problem of finding QoS fulfilling paths in unicast and multicast scenarios. . . . 5
2.2 Routing with incomplete information - the model . . . 7
2.2.1 Finding optimal paths in the unicast scenario . . . 8
2.2.2 Finding optimal trees in the multicast scenario . . . 8
2.2.3 Optimizing the link scaling - Signaling bandwidth versus routing performance . . . 9
2.3 Solution for the unicast routing problem . . . 10
2.3.1 Solution for the bottleneck routing problem in unicast case . . . 10
2.3.2 Novel algorithms for ARII . . . 11
2.4 Multicast routing in WSN applications with incomplete information . . . 19
2.4.1 Bottleneck type requirement . . . 21
2.4.2 End-to-end additive requirement . . . 22
2.4.3 Approximation by HNN . . . 24
2.5 Optimal link scaling as a constrained optimization problem . . . 27
2.5.1 Traffic modeling and queuing system . . . 28
2.6 Simulations and numerical results . . . 32
2.6.1 Routing in unicast case . . . 32
2.6.2 Multicast routing with incomplete information . . . 37
2.6.3 Link scaling . . . 38
2.7 Conclusion . . . 42

3 Thesis group II - a heuristic solver based on hypergraphs for UBQP and its applicability in ICT 43
3.1 Related work . . . 44
3.2 UBQP formulation . . . 45
3.2.1 Successive reduction methods for solving the UBQP . . . 45
3.2.2 Breaking down UBQP into smaller dimensional sub problems . . . 47
3.3 Novel approaches to UBQP based on dimension reduction . . . 50
3.4 Performance analysis of the novel algorithms . . . 54
3.5 Application of UBQP to Scheduling . . . 60
3.6 Application of UBQP to MUD in an environment with fading . . . 65
3.7 Conclusion . . . 69

DOI:10.15774/PPKE.ITK.2019.006

4 Thesis group III - near Bayesian performance non-parametric detection with Feed Forward Neural Networks 70
4.1 Non parametric approach to Multiuser Detection - Optimal decision as a maximum search problem . . . 71
4.2 Application of the FFNN as an optimal detector . . . 72
4.3 New coding techniques for minimum complexity FFNN . . . 73
4.4 Numerical performance analysis . . . 77
4.4.1 Network architecture and training parameters . . . 78
4.4.2 Performance of the new detector . . . 79
4.5 Conclusions . . . 82

5 Summary of the dissertation and closing remarks 83

Appendices 84

A New scientific results and theses of the dissertation 84

B Artificial Neural Networks outline 89
B.1 Hopfield Neural Network (HNN) . . . 89
B.2 Feed Forward Neural Network (FFNN) . . . 90
B.3 An outline of deep learning - perspective for neural networks . . . 91

C Notation and general assumptions 92
C.1 Notation of graphs . . . 92
C.2 Notation of sets, ordered sets (series), matrices, and vectors . . . 92
C.3 Notation of subsets and elements of matrices and vectors . . . 92
C.4 General assumptions on UBQP problems . . . 93

D A word on the log-moment generating function 94

E Short description of the GSMT and CGSMT problem 97
E.1 The GSMT problem . . . 97
E.2 The CGSMT problem . . . 97

F Description and performance characterization of the new UBQP related algorithms 99
F.1 Dimension reduction of the greedy algorithm - L01 . . . 99
F.2 Dimension reduction of the first chance algorithm - D01 . . . 99
F.3 The description of dimension adder DA01 algorithm . . . 100
F.4 The description of the dimension adder DA02 algorithm . . . 101
F.5 Run-time and performance analysis tables of the UBQP solvers . . . 101

G List of figures, tables and algorithms 104

DOI:10.15774/PPKE.ITK.2019.006

Acronyms

ACK acknowledgment packet 19
ARII Additive Routing with Incomplete Information iv, 8, 11–14, 81
AS Autonomous System 6
ATM Asynchronous Transfer Mode 6

BB Branch and Bound 41
BER Bit Error Rate viii, ix, 65, 67, 73–76, 79
BM Boltzmann Machine 91
BQP Binary Quadratic Programming 4, 50, 63, 65, 85

CDF cumulative distribution function 7
CDMA Code Division Multiple Access viii, ix, 61, 62, 66
CGSMT Constrained Graph Steiner Minimal Tree v, 2, 4, 6, 22, 26, 83, 97, 98, 103
COP Combinatorial Optimization Problem 41, 42
CP Cutting Plane 41
CSP Car Sequencing Problem 41

DBN Deep Belief Network 91
DDT Devour Digest Tidy-up viii, ix, 41, 52
DF Decision Feedback 62
DHNN Discrete Hopfield Neural Network 22, 23, 41, 59, 88, 99–101, 103

EA Evolutionary Algorithm 41
EDA Estimation of Distribution Algorithm 41

FCFS First-Come-First-Served 28
FFNN Feed Forward Neural Network v, viii, ix, 3, 4, 66–71, 73, 74, 76, 79, 86, 87, 89–91, 103
FH Frequency Hopping 62

GA Genetic Algorithm 41, 44, 66
GPP Graph Partitioning Problem 41
GSMT Graph Steiner Minimal Tree v, 6, 18, 20, 97, 98

HNN Hopfield Neural Network iv, v, viii, ix, 7, 18, 20, 21, 23, 26, 39, 41, 51, 52, 54, 59, 60, 62, 83, 85, 87, 88, 91, 103

IaaS Infrastructure as a Service 40
ICT InfoCommunication Technology iv, ix, 40, 51, 80, 85
IoT Internet of Things viii, ix, 1, 5, 40, 56, 80
IP Internet Protocol 1, 5, 6
ISI Inter Symbol Interference 61, 74
ITS Iterated Taboo Search 41

LAS Link Advertisement Scheme 2, 7, 12, 13, 81, 87, 102
LE Link Entropy 2, 9, 26, 27, 30, 36–39, 83, 102
LS Local Search viii, ix, 41, 43, 44, 51, 52
LSA Link State Advertisement 9, 87, 102

MA Memetic Algorithm 41, 44
MAP Markovian Arrival Process 27, 28, 30, 31, 83
MAP Maximum A-Posterior 28, 66–71, 73, 75, 76, 79
MBER Minimum Bit Error Rate 66
MIMO Multiple In Multiple Out 62, 66
MIQP Mixed Integer Quadratic Programming 41
ML Maximum Likelihood 62, 67, 74
MLD Maximum Likelihood Detector 62
MLP Most Likely Path 8, 10–12, 17
MMPP Markov Modulated Poisson Process 27, 36
MMSE Minimum Mean Square Error 66, 74, 76
MPLS Multi-protocol Label Switching 6
MST Minimal Spanning Tree 97
MUD Multiuser Detection iv, v, viii, ix, 2–4, 40, 51, 61, 62, 65–67, 73, 80, 85, 86
MUI Multi User Interference 61

NN Neural Network 66

DOI:10.15774/PPKE.ITK.2019.006

NP Non-deterministic Polynomial-time viii, ix, 26, 40, 41, 83, 88

OFDM Orthogonal Frequency Division Multiplexing 66
OFDMA Orthogonal Frequency Division Multiple Access 62
OLS Optimal Link Scaling 5, 7
ORLIB Operation Research Library viii, ix, 40, 52
OSI Open Systems Interconnection 1, 5, 6
OSPF Open Shortest Path First 6
QOSPF Quality Open Shortest Path First 6

PBF Pseudo Boolean Function 41
PDF probability density function ix, 12
PHY Physical layer 4, 61
PMF probability mass function 16
PNNI Private Network-to-Network Interface 6

QAP Quadratic Assignment Problem 41
QBD Quasi Birth-Death 27, 28, 30
QoS Quality of Service iv, viii, ix, 1, 2, 4–8, 11, 14–16, 18, 20, 31, 39, 80, 81
QP Quadratic Programming 41

RBFN Radial Basis Function Network 66
RBM Restricted Boltzmann Machine 91
RNN Recurrent Neural Network 66, 87, 91

SA Simulated Annealing 41, 44
SB Signaling Bandwidth 26
SD Sphere Detector 62, 74, 76
SDMA Space Division Multiple Access 66
SDR Semi Definite Relaxing viii, ix, 39, 41, 52, 53
SE Signaling Entropy viii, ix, 2, 9, 26, 30, 31, 36–39, 66, 83, 102
SMT Steiner Minimal Tree 97
SNR Signal to Noise Ratio 74, 76, 77, 79, 103
SONN Self Organizing Neural Network 41
SPR Shortest Path Routing 8, 10, 13–15, 17, 81, 82
SS Scatter Search 41
SS-MC-MA Spread Spectrum Multi Carrier Multiple Access 62

TDMA Time Division Multiple Access 56
THP MIMO Tomlinson Harashima Precoded Multiple Input Multiple Output 62
TLU Threshold Logic Unit 87
TS Taboo Search viii, ix, 41, 44, 59, 61
TSP Traveling Salesman Problem 41
TWT Total Weighted Tardiness 58–60, 65, 103

UBQP Unconstrained Binary Quadratic Programming iv, v, viii, ix, 2, 3, 40–44, 46–48, 50–52, 56–59, 61, 62, 65, 67, 74, 80, 85, 88, 93, 102, 103
UWB Ultra Wide Band 62

WSN Wireless Sensor Network iv, viii, 1, 5, 9, 18, 21, 56

ZF Zero Forcing 74, 76

DOI:10.15774/PPKE.ITK.2019.006

Kivonat

Új, neurális hálózat alapú algoritmusokat mutatok be infokommunikációs környezetben előforduló NP-nehéz problémák szuboptimális megoldására. A disszertáció három fő tématerületet érint: előírt minőséget (QoS-t) biztosítani képes útvonalválasztó algoritmusokat unicast és multicast esetekre, melyek tipikusan IoT, WSN és streaming alkalmazásokban fordulnak elő; problémákat, melyek megfogalmazhatóak diszkrét kvadratikus optimalizálási feladatként (UBQP), úgy mint ütemezési, skálázási és többfelhasználós detekciós (MUD) feladatokat elosztott kommunikációs rendszerekben. Illetve egy általános mintakeresési eljárást zajos és torzított adatokra.

A QoS-t biztosítani képes algoritmusoknak olyan hálózatokban van jelentősége, ahol a hálózat állapota nem ismerhető meg teljesen. Ez a bizonytalanság egyrészt a forgalom véletlenszerű fluktuációjából, másrészt a hierarchikusan szerveződő protokollok információ aggregálásából származik. Ezt a bizonytalanságot az általam javasolt algoritmusok valószínűségi változók segítségével beépítik a modelljükbe. Ennek következtében olyan útvonalakat keresnek, melyek maximális valószínűséggel teljesítik az adott QoS kritériumot (pl. végpontok közti késleltetés). Az új algoritmusokat mind unicast, mind multicast útvonalválasztás esetre bemutatom. A modell feltételezi, hogy a link leírókat modellező valószínűségi változók vagy gaussi eloszlásúak, vagy a nagy eltérések elmélete alapján modellezhetőek, így választják ki az optimális útvonalakat. A javasolt algoritmusok képesek optimális(unicast) illetve szuboptimális(multicast) útvonalak megtalálására polinomiális időben, miközben az előírt QoS kritériumot teljesítik. Továbbá a jelzési folyamatok optimalizálását is bemutatom információ elméleti mértékek segítségével.

A kvadratikus optimalizálási problémákra (UBQP) hipergráf alapú reprezentáción, neurális hálózatok által kezelhető dimenzió csökkentő és növelő algoritmusokat definiálok. Ezzel a módszerrel hatékonyan kereshető jó minőségű szuboptimális megoldás. A javasolt algoritmusok közvetlenül alkalmazhatóak jelen kommunikációs technológiai problémákra, mint például “cloud computing”-beli ütemezési vagy skálázási problémákra illetve többfelhasználós detekciós problémákra (MUD). Az ütemezési probléma esetén a javasolt algoritmusok az összehasonlításban jobb “Weighted Tardiness" értékeket érnek el, míg a MUD esetén a bithiba arány (BER) megközelíti az elméleti határt. Az algoritmusok teljesítőképességét szintén összevetem tradicionális algoritmusokkal (DDT, HNN, LS, TS és SDR) illetve megvizsgálom egy referencia problémahalmazon is (ORLIB), mely igazolja hogy az új algoritmusok jobban teljesítenek a hasonló komplexitású algoritmusokhoz képest.

A mintakeresési feladat megoldására egy lineáris kódolási eljárást javasolok, mellyel egy előrecsatolt neurális hálózat(FFNN) számára készíthető tanulóhalmaz. Az eljárással nem-parametrikus módon lehet mintaillesztést végezni, melyet a CDMA rendszerekben előforduló, ismert MUD problémán mutatok be. Ezzel együtt az eljárás könnyen kiterjeszthető más mintakeresési feladatra is. Az új kódolási eljárás egyrészt növeli a kommunikációs hálózat áteresztő képességét, miközben a neurális hálózat komplexitását is csökkenti. Bizonyítom, hogy aszimptotikusan optimális teljesítmény érhető el a javasolt algoritmussal, ami kihasználható a spektrális hatékonyság (SE) növelésére. Az állításokat szimulációval támasztom alá, melyben a javasolt algoritmusok közel teljesítenek az elméleti optimumhoz valós csatornamodellekre (COST-207).

DOI:10.15774/PPKE.ITK.2019.006

Abstract

Novel neural network based algorithms are introduced for approximating the solutions of NP-hard discrete optimization problems in InfoCommunication Technology (ICT). Three main topics are addressed: QoS providing routing strategies in unicast and multicast scenarios typically found in IoT and streaming applications; problems that can be translated into UBQP including scheduling, MUD, scaling in distributed computing environments; and general Bayesian pattern matching for noisy and distorted data.

The QoS aware routing algorithms are for networks where link states are characterized by incomplete information. Incompleteness in link state can either be due to random traffic fluctuations or to aggregation in link description because of hierarchical protocols. This incompleteness is taken into account by characterizing the link states with random variables subject to a certain PDF. As a result, routing amounts to seeking paths satisfying a given end-to-end QoS requirement (e.g. end-to-end delay) with maximal probability. Novel algorithms are proposed to provide optimal paths satisfying given end-to-end requirements with maximal probability in the case of single- and multicast routing. The proposed algorithms are based on either assuming Gaussian link delay distribution or using large deviation theory to find the most likely path. The proposed methods are capable of QoS routing in polynomial time. Furthermore the optimization of the in-band signaling is taken into consideration by modeling it with entropy-like quantities.

For the quadratic optimization problems - commonly referred to as UBQPs - the proposed methods are based on hypergraph representation and recursive dimension reduction or addition of the search space. In this way, efficient and fast search can be carried out and high quality sub-optimal solutions can be obtained in real-time. The new algorithms can directly be applied to the problems of present day communication technologies, such as scheduling in cloud computing environments or MUD for improved performance. In the case of scheduling better Weighted Tardiness can be achieved by running the proposed algorithms while in the case of MUD, the achieved BER can approximate the Bayesian optimum. The methods are also tested on large scale quadratic problems selected from ORLIB and the solutions are compared to the ones obtained by traditional algorithms, such as DDT, HNN, LS, TS and SDR. As the corresponding performance analysis reveals the proposed methods can perform better than the traditional ones with similar complexity.

For the general pattern detection problem I propose linear coding techniques for implementing non-parametric neural network based detectors. I present it on the well known MUD problem in CDMA systems, however it can be easily extended to general pattern detection problems. These new encoding schemes on one hand can increase the data speed over the channel and reduce the complexity of the FFNN based detector on the other. It is proven that asymptotically optimal detection performance can be achieved by the proposed algorithms. It will be also demonstrated that the data rate can be increased and the complexity of the corresponding neural network at the receiver side can be decreased by the novel coding schemes. This allows us to improve SE as well while maintaining the performance of the methods. Extensive simulations demonstrate that the performance of the proposed algorithms are near optimal on real channel models (COST-207).

DOI:10.15774/PPKE.ITK.2019.006

1 Introduction

Due to recent and vast improvement in sensor technology and to the yet unfailing trend set by Moore's law, multitude of novel fields opened up for new applications. The platform carrying these novel applications are at the same time required to support mobility and flexibility at the end user side. As the applications become evermore complex their supporting systems have to deal with more demands. These (such as their own communication networking subsystem, central processing units or the devices in the background network which they communicate with) have to act more intelligently and adapt to the new demands arriving from the upper layers. Also the majority of the end user devices are portable and use battery as a power source, it becomes increasingly important to take this into consideration at the widest range of system design possible.

Typically into these areas we could count in the IoT based applications, monitoring and intervening systems that are based on WSNs [2], peer-to-peer and "broadcast" type relaying and processing systems dealing with multimedia streams (let that be video or audio) or most of the cloud based services [136]. These services and systems have the common aspect of providing a certain QoS, while their resources are time and location dependent and also limited [157]. On inherent shared resources like on the radio subsystem these constraints appear even more stringent. Furthermore scheduling tasks efficiently in these distributed environments are imperative.

Networking technologies used today are following a layered structure [84] and most of them form packet switched communication networks. In these networks typically there are no separate resource dedicated for signaling and controlling, but without these processes providing a required QoS can be mostly done in a best effort manner if possible at all. Requiring these crucial processes to be present means that they have to isolate an additional portion from the resources bearing the payload to themselves. Thus the total capacity from the end user perspective further diminishes in contrast to their signaling-less counterparts. The best effort like structures which build up the traditional packet switched networks do not or rarely use signaling to govern their internal mechanisms in respect to QoS. These structures consist of protocols defined by OSI in several layers. Examples for these in the lower OSI layers (physical, data-link, network) are the Ethernet [1], the 802.11 [82] or the IP. Although their design does not incorporate providing QoS directly [112, 90], they are widespread because of their reliability and usability [90]. One of the base questions of the packet switched networks is to find an appropriate path and scheduling for the data packets to traverse through the network from the source to the destination. [131, 90, 107, 19, 18]. Applications using wireless technologies in the physical layer face even more constraints due to the shared nature of the radio media, which has to be accounted for if one requires to provide QoS.

In the light of this the open questions that this dissertation aims to answer are:

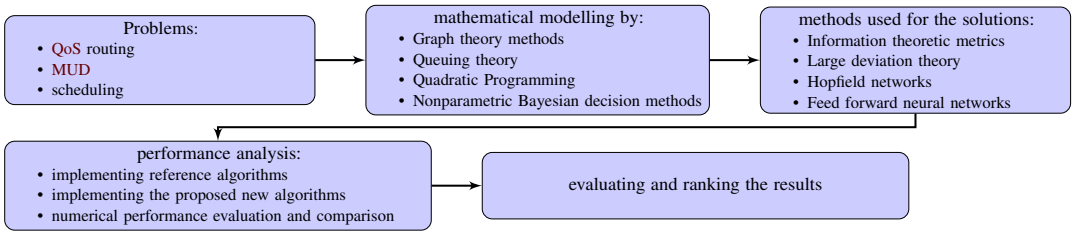
- How can one find an appropriate path in a packet switched network which provides a QoS. (QoS unicast, multicast routing)
- How can one perform scheduling tasks in communication networks efficiently with algorithms which lend themselves to parallelization. (scheduling)
- How can one solve at the physical layer which use wireless technologies near optimally the

DOI:10.15774/PPKE.ITK.2019.006

MUD problem in a parallelized fashion. (Multiuser Detection)

1.1 Applied methodology

This dissertation uses the following general approach to investigate the problems and bring up possible solutions as theses:



The summary of the problems, the used algorithms, theories and used tools can be found at Table 1.

technological challenges	*model formulation and problem category*	*theoretical performance*	*used methods and algorithms*	*related theses*
Unicast routing	using random link descriptors and LAS reducing the problem to additive and bottleneck type metrics	The path of choice is capable of providing a QoS	Queuing models , Markov modulated Poisson distributions, Gaussian approximation, Large deviation theory	Thesis I.1 Thesis I.2
Optimizing LAS	Applying information theoretic metrics (Link Entropy and Signaling Entropy) as constrains in optimization	Using the resulting LAS the bandwidth of the signaling process in the network can be bound and kept under a predefined value while at the same time the uncertainty of the link states in the network is also kept under a well defined value.	Information theoretic measures, exhaustive search, general nonlinear constrained optimization methods	Thesis I.4
Multicast routing	using random link descriptors search for a CGSMT in the network and reformulating the search of the CGSMT as an UBQP	Sending streams through the paths from the source and destination points provide the prescribed QoS, while they perform optimally in the sense that they strain the chosen resource minimally. These constraints posed on the resources are like minimal energy consumption or minimal total used bandwidth to deliver the payload.	Gaussian approximation, large deviation theory, Hopfield network, binary quadratic programming methods	Thesis I.3

- continuing on the next page -

DOI:10.15774/PPKE.ITK.2019.006

Table 1 – continued from previous page –

technological challenges	*model formulation and problem category*	*theoretical performance*	*used methods and algorithms*	*related theses*
Multiuser Detection (MUD)	UBQP and non-parametric Bayesian detection	The exact solution of the UBQP is the optimal solution of the problem. The proposed UBQP "solvers" lend themselves to parallelization well, since they are made up from simple nonlinear computing units. They give well performing sub-optimal solutions with fast convergence time. In case of the non-parametric models the proposed algorithm can arbitrarily approximate the optimal decision at the expense of increasing its complexity.	Algorithms operating on a hypergraph based dimension reduction and dimension addition, Hopfield network, binary quadratic programming methods, Feed Forward Neural Network (FFNN), logarithmic search	Thesis II.1 Thesis III.1
Scheduling	UBQP	Exact solution to the UBQP is the optimal solution to the problem. The proposed "solvers" give sub-optimal solutions at fast convergence speeds.	Hopfield networks, binary quadratic programming methods	Thesis II.1

Table 1: Problems investigated, used algorithmic tools and the related theses

1.2 Numerical analysis framework

Throughout the dissertation several numerical results are presented. These results were obtained by using computer simulations. The framework in which the algorithms were evaluated was written by the author mostly in the Matlab™ programming language and was evaluated in the same runtime environment.

The codebase was almost entirely implemented by the author. It includes the tools used for the mathematical modeling, the reference and the newly developed algorithms, the communication network simulation framework and the neural network implementations. For consistent visual presentation the figures were also produced in the same environment. I license the codebase under creative commons license "Attribution-NonCommercial" (CC BY-NC), thus can be used freely for non-commercial purposes and distribute derivatives under the same license. A copy can be requested from the author or can be found with the hard printed version of the dissertation. Although 3rd party toolboxes were also used and referenced, they are also freely available and can be used according to their own copyrights.

Three main frameworks were written corresponding to the three main thesis groups that this document presents. Also note that the codebase is the result of several years of work during which the author gained experience and knowledge about bad and better programming patterns and solutions. Consequently the code is neither consistent in terms of used methodology nor thought to be error free, but validates the predictions of the theoretical models.

DOI:10.15774/PPKE.ITK.2019.006

1.3 Structure of the dissertation

This dissertation presents the problems and the theses in three groups in three sections.

I) In the first group at section 2, I elaborate on the route searching problem providing QoS in packet switched networks both for unicast and multicast cases and propose novel methods to solve or approximate them. In this section I also investigate the link scaling problem and propose a method for the optimization using information theoretical measures.
The theses in this group can be found:
   - Thesis I.1 on page 13
   - Thesis I.2 on page 19
   - Thesis I.3 on page 27
   - Thesis I.4 on page 31

II) In the second group at section 3, I elaborate on the Multiuser Detection (MUD) and scheduling problems in communication technologies. I formulate these tasks as Binary Quadratic Programming (BQP) and I propose a family of algorithms which act as a sub-optimal solver to the BQP problem. These algorithms lend themselves to parallelization well because of their inherent structure.
Thesis II.1 can be found on page 54

III) In the third group at section 4, I propose an FFNN based algorithm which performs comparably to the non-parametric optimal Bayesian decision for the MUD problem defined for the wireless networks PHY layer.
Thesis III.1 can be found on page 77.

Each thesis group has the following structure:

- A short introductory section where the base problems and the open questions are posed.
- Introducing the models for the problems.
- Elaborating on the models then making statements of the theses in the thesis group.
- Investigating the performance of the proposed methods through applications.

At the end of the dissertation in section 5 conclusions are drawn and possible extensions and development directions are named.

In Appendix A the theses are stated in a self consistent manner for easier overview.

The details of the proposed algorithms, description of the CGSMT problem and a brief overview of the used neural networks were also moved to the appendices to not draw attention away from the main course of discussion, however they are integral part of this document. The brief overview of the neural networks aims to summarize the common modes how these networks are used in this document and to emphasize their usability in modern day communication systems.

DOI:10.15774/PPKE.ITK.2019.006

2 Thesis group I - routing with incomplete information in unicast and multicast scenarios

In this thesis group I propose novel solutions to the problem of unicast and multicast QoS routing with incomplete information. These new methods are capable of finding a sub-optimal path and a multicast route set in polynomial time. Also in this thesis group I provide a solution to the OLS task by using information theoretical measures, such as "signaling entropy" and "link entropy". *The corresponding publication of the author is titled "Multicast Routing in Wireless Sensor Networks with Incomplete Information"[161]*

2.1 Problem of finding QoS fulfilling paths in unicast and multicast scenarios.

In networks where is no dedicated channel for signaling or controlling purposes (typical in IoT, networks using IP) these procedures consume additional resources. As a result it diminishes the capacity available for information transfer, however at the same time services heavily demand the speed and reliability of the underlying communication stack. This gives rise to the problem of finding QoS fulfilling paths. For example in the IoT vision every device can send and receive information and might act as an intermediate node. From an angle these devices can be seen forming a WSN. If the network contains battery operated devices then the applications also require reliable communication while keeping energy consumption at a minimal level (e.g. consider a smart home application where the user should not be forced to change the batteries frequently or a smart agricultural application where battery change might not be feasible at all). On the other hand in an application where the energy consumption might not be a problem (e.g. a smart fridge, automotive application or a factory with smart production appliances) other types of reliability criteria exist that the (sub)networks must meet. Among others, these can be robustness to communication shortage, redundancy, efficient use of the communication bandwidth, responsiveness, etc. One can also consider peer-to-peer applications, e.g. video on demand services or voice over IP services, where large quantity of data needs to be reliably transported to the peers. In these scenarios both unicast and multicast type communication is common. Data is to be transmitted to a single or a set of destination nodes with the packets routed in a multi-hop manner where intermediate nodes are also used for packet forwarding.

Legacy network structures which were not designed for QoS but mostly for best effort usually have no dedicated channel for signaling. Examples of these protocols in the lower OSI layers (physical, data-link, network) are the Ethernet, 802.11 or IP. Despite they can not provide QoS natively by their design[112, 90], they are widespread because of their reliability and interoperability[90]. Nevertheless, there is a need to run QoS communication over these best effort platforms. One of the major challenges in IP networking is to ensure QoS routing, which selects paths to fulfill end-to-end delay or bandwidth requirements [131, 90, 107, 19, 18] as opposed to traditional shortest path routing. Because of the manifold optimization criteria[85], even in the unicast case QoS routing can prove to be much harder than the problem of finding optimal paths based on merely the hop count [85]. Protocols striving to provide QoS need to have some knowledge about the state of the network. This information has to be propagated also (signaling). QoS-aware routing protocols often propose a method to reduce the amount of state

information that have to be kept synchronized among routers, called topology aggregation (e.g. in: QOSPF, PNNI). Thus, routing information describing a certain domain of the network have to be aggregated, which acts as another source of uncertainty of resource availability information. Another source of uncertainty is introduced when multiple hierarchical levels are connected. Such a pattern is inevitable from the base rules of the network design. These hierarchical levels are introduced because these networks connects through inter-domains.

Incorporating QoS into routing has been long studied[90]. For example an extension to the classical IP over ATM system to support application level QoS is studied in [112]. Also QoS aware routing algorithms exist in many levels, such as in inter-domain level [89], in MPLS based network parts [111], inside ASs [131], but routing is done mostly by a hierarchical manner. In networks following the OSI model the routing is done in the 3rd (network) layer. The OSPF routing protocol offers two, while PNNI offers many levels, where routing can be performed in a hierarchical manner [55]. Subnetworks on a given level of the hierarchy are abstracted as "nodes" for a higher layer and delay information in those subnetworks are aggregated into an average QoS parameter. On the other hand, randomly fluctuating traffic load on links can also result in random delays. Thus link delays are periodically advertised when the delay surpasses a given threshold (e.g. in PNNI and QOSPF standards, see [3, 55]). These thresholds are defined in advance. This prompts us to take delays into account as random variables characterized by their probability distribution functions over the interval between two reported thresholds [58, 146, 103]. The distribution of these thresholds (and therefore the length of the intervals over which the link delay is not fully characterized) can be equidistant or non-equidistant. In practice non-equidistant thresholds are used, since in this case the impact on network utilization by sending only signaling information (part of the available bandwidth is used for carrying information about link delays) is minimized.

The phenomena described above give rise to the task of routing with incomplete information. Namely, how to select paths to fulfill end-to-end delay requirements in the lack of the exact values of link delays [58, 19, 18]. Routing is then perceived as an optimization problem to search over different quality paths, where the quality of a path is determined by the probability of meeting the end-to-end QoS requirement [99, 58]. Unfortunately, routing with incomplete information in general cannot be reduced to the well-known Shortest Path Routing (SPR), thus it cannot be solved in polynomial time in general.

The multicast scenario can be seen as an extension of the previous problem and can be treated as the well-known GSMT problem, which has proved to be NP-hard even for deterministic link descriptors and cost functions[93]. The CGSMT problem is even more difficult, where the minimal cost tree is sought by guaranteeing a given QoS at the same time. This has proved to be NP-hard as well[94]. On the other hand, common multicast routing algorithms utilize stored network state information[87], which can quickly become obsolete due to the changing fading radio environment or traffic pattern. Link state information in clustered networks can also be incomplete due to aggregation. Heuristic algorithms for finding Multicast Trees are published in [144], however the computational complexity becomes overwhelming as the number of nodes increases in the network. That is why, we would like to benefit from the fast optimization properties of the

DOI:10.15774/PPKE.ITK.2019.006

HNN [130]. The execution time of such algorithms only depends on the interconnectivity of the network because every neuron represents an edge in the graph.

The QoS over an obtained path, however strongly depends on the “incompleteness” of link descriptors which is determined by the thresholds initiating Link State Advertisement (LSA)s. These thresholds are referred to as a Link Advertisement Scheme (LAS). The process of defining LASs over the network is called link scaling, which can be subject to further optimization in order to economize on signaling bandwidth. The smaller these intervals are, the smaller is the measure of “incompleteness” under which a path is selected. As a result, the routing performance is higher. On the other hand, small intervals forces more frequent announcements of the values of the link descriptors throughout the network, which implies that considerable portion of bandwidth is used up for transmitting signaling information. Thus, the optimization of the size of these intervals is a crucial task for network management. This problem is referred to as OLS. As can be seen, OLS is a constrained optimization problem, in which one has to maximize routing performance under the constraint of keeping the signaling bandwidth bellow a given threshold.

2.2 Routing with incomplete information - the model

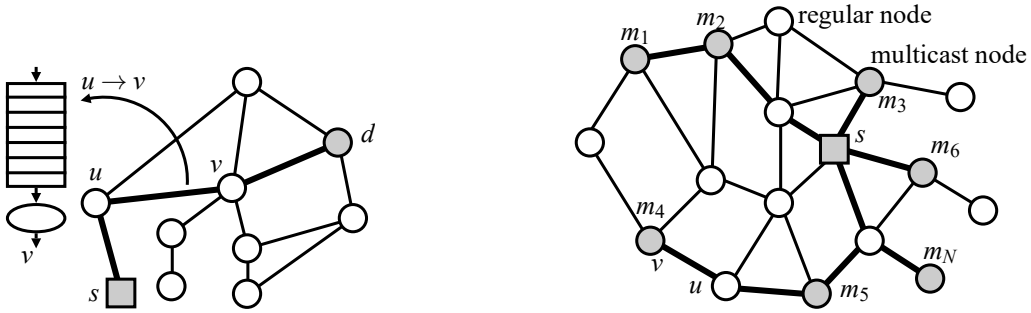
Because of the aforementioned reasons the state of the traffic link called link descriptor is usually modeled by a random variable [58, 146]. This gives rise to the extension of the traditional minimum-hop based routing problem to a procedure where statistical and information theoretical metrics has to be also considered. Consequently routing becomes a task to find a path or set of paths on which the data flow satisfy the predefined QoS criteria with a certain probability[58, 146, 145, 99, 59].

Therefore the communication network is modeled by a graph

$$G \; V \; E \quad _{u\,v} \quad F_{u\,v} \quad x \tag{2.1}$$

- where nodes are denoted by index $u \quad V$ and links referred to as node pairs $u \; v \quad E$
- each link $u \; v \quad E$ has a QoS link descriptor $_{u\,v}$ which is assumed to be a random variable subject to a CDF $F_{u\,v} \; x \quad _{u\,v} \quad x$
- random variable $_{u\,v}$ is referred to as "bottleneck measure" if the smallest link measure determine the quality of the route ($_{u\,v}$ is a "bandwidth-like" variable); It is also referred to as "bottleneck measure" if the largest link measure determine the quality of the route ($_{u\,v}$ is a "energy consumption-like" variable)
- random variable $_{u\,v}$ is referred to as "additive measure" if the sum of link measures contained in the path determine the quality of the path ($_{u\,v}$ is a "delay-like" variable)

DOI:10.15774/PPKE.ITK.2019.006



(a) unicast case (b) multicast case

Figure 1: Graph model of the network

2.2.1 Finding optimal paths in the unicast scenario

The objective is to find an optimal path $\tilde{R}$ from all possible paths $^{s\ d}$, which most likely fulfills the given QoS criterion, namely:

$$\tilde{R} \quad \underset{R \quad s\ d}{\text{argmax}} \quad \min_{u\,v \quad R} \quad {}_{u\,v} \quad \text{B} \tag{2.2}$$

in the case of a bottleneck measure or

$$\tilde{R} \quad \underset{R \quad s\ d}{\text{argmax}} \quad \sum_{u\,v \quad R} \quad {}_{u\,v} \quad \text{T} \tag{2.3}$$

in the case of an additive measure. The path $\tilde{R}$, introduced above, will be referred to as the Most Likely Path (MLP). In case of having additive link measures, I will refer to this problem as Additive Routing with Incomplete Information (ARII). It is well known that Shortest Path Routing (SPR) can be solved in polynomial complexity by Dijkstra or Bellman-Ford algorithms. Therefore, mapping an MLP problem into an SPR is equivalent with proving that MLP can be solved in polynomial time. The corresponding link measure on which basis the SPR algorithm selects the shortest path is, in general, denoted by $_{u\,v}$ $u\ v$ E. As a result, our aim is to prove that under certain circumstances the search for MLP can be done by an SPR algorithm by finding the appropriate mapping $_{u\,v}$ $_{u\,v}$.

2.2.2 Finding optimal trees in the multicast scenario

In this case the information source, typically a Base Station, is denoted by src V and the set of multicast destination nodes by M $m_1\ m_2$ m_N V. In this case the objective is to find an optimal tree $\tilde{\text{A}}$ from all multicast trees $^{src\ M}$ which most likely fulfills the given QoS criterion, namely:

$$\tilde{\text{A}}_1 \quad \underset{\text{A} \quad src\ M}{\text{argmax}} \quad \max_{u\,v \quad \text{A}} \quad {}_{u\,v} \quad \text{P} \tag{2.4}$$

for the bottleneck type measure or

$$\tilde{\text{A}}_2 \quad \underset{\text{A} \quad src\ M}{\text{argmax}} \quad \max_{R_{src\,m} \quad \text{A}} \quad \sum_{u\,v \quad R_{src\,m}} \quad {}_{u\,v} \quad \text{T} \tag{2.5}$$

DOI:10.15774/PPKE.ITK.2019.006

for the additive type measure. Note that for the multicast bottleneck case the problem is formulated differently than in the unicast case, because the link descriptor is typically power consumption like quantity in WSNs, thus the bottleneck in this case is the most consuming element. But the derived conclusions are also applicable to a bandwidth-like link descriptor with minor modifications.

2.2.3 Optimizing the link scaling - Signaling bandwidth versus routing performance

An advertisement of a link state change happen when the current value of link descriptor steps out from an interval defined by two threshold values, which is called Link State Advertisement (LSA). Figure 2 depicts an LSA in case the link descriptor is of delay type. When a link advertise-

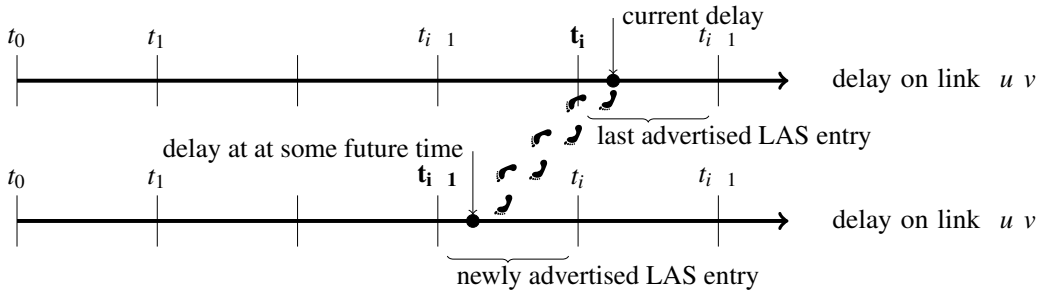


Figure 2: LSA in case the link descriptor is of delay type.

ment is received, the receiver could assume that the link descriptor of the sender node is within the new interval. The finer the threshold grid is defined on the link descriptors the more precise the receiver can be about the current state of the sender, but on the other hand it also means the frequency of the advertisement to increase. Here our objective is to strike an optimal balance between the signaling bandwidth and routing performance. To capture the underlying phenomena two information theoretical measures are introduced: the *Signaling Entropy (SE)* and the *Link Entropy (LE)*.

Since link state advertisement occurs when randomly jumping over one (or several) advertisement threshold, it can be regarded as an information theoretical source denoted by $_{u\,v}$ with values $t_0\ t_1$ $1_{Z\ 1}$ covering the axis of the link descriptor with probabilities $\hat{p}_0\ \hat{p}_1$ $\hat{p}_{Z\ 1}$. The source does not emit any symbol (inactive) with probability $\hat{P}$ 1 $\overset{Z\ 1}{_{i\ 0}}\hat{p}_i$. These probabilities are governed by the randomly fluctuating link traffic (i.e. by the varying queue length in the buffer as each link is perceived as a buffer). Assuming optimal source coding, the necessary bandwidth to distribute the link state information is related to the conditional entropy H $_{u\,v}$ $_{u\,v}$ 1 of $_{u\,v}$, where the Bernoulli random variable $_{u\,v}$ describes whether the source is active or not. The link entropy on the other hand is the measure which quantifies the uncertainty about state of link $u\ v$ if $t_i\ t_{i\ 1}$ interval was advertised for this link ($_{u\,v}$ i). Using these two quantities the bandwidth and the quality of the signaling processes can be kept at bay.

link entropy: H $_{u\,v}$ $_{u\,v}$ (2.6a)

signaling entropy: H $_{u\,v}$ $_{u\,v}$ (2.6b)

DOI:10.15774/PPKE.ITK.2019.006

Note that both $_{u\,v}$ and $_{u\,v}$ depends on the choice of the LAS. The underlying question is, how can we choose a LAS, so that the signaling bandwidth does not exceed a given threshold and at the same time the link entropy (the uncertainty about the state of the network) is minimal.

2.3 Solution for the unicast routing problem

The main contribution of this section is that I provide the necessary conditions and the appropriate mappings $_{u\,v}$ $_{u\,v}$ for finding an MLP in case of having bottleneck or additive link measures. Using the mapping the routing task can be solved by an SPR algorithm.

2.3.1 Solution for the bottleneck routing problem in unicast case

The following lemma establishes that an MLP with bottleneck measure can easily be solved by using traditional SPR algorithms.

Lemma 1. *The solution of*

$$\tilde{R} \quad \underset{R}{\operatorname{argmax}} \quad \min_{u\,v \quad R} \quad _{u\,v} \quad \mathrm{B} \qquad \text{(2.2 revisited)}$$

is equivalent to solving a traditional shortest path problem with the metric assigned to link $u\ v$ *being* $_{u\,v}$ $\ln$ $_{u\,v}$ B *if the link descriptors are independent.*

Proof. We seek the path

$$\tilde{R} \quad \underset{R}{\operatorname{argmax}} \quad \min_{u\,v \quad R} \quad _{u\,v} \quad \mathrm{B} \qquad \text{(2.2 revisited)}$$

which is equivalent to

$$\tilde{R} \quad \underset{R}{\operatorname{argmax}} \quad \underset{u\,v \quad R}{\quad} \quad _{u\,v} \quad \mathrm{B} \qquad (2.7)$$

If $_{u\,v}$ -s are independent, then

$$\underset{u\,v \quad R}{\quad} \quad ^{u\,v} \quad \mathrm{B} \quad \underset{u\,v \quad R}{\quad} \quad _{u\,v} \quad \mathrm{B} \qquad (2.8)$$

thus one can write

$$\tilde{R} \quad \underset{R}{\operatorname{argmin}} \quad \underset{u\,v \quad R}{\quad} \quad \log \quad _{u\,v} \quad \mathrm{B} \qquad (2.9)$$

Therefore assigning measure to link $u\ v$ as $_{u\,v}$: $\log$ $_{u\,v}$ B , MLP routing can indeed be solved by SPR. □

Note that similarly

$$\tilde{R} \quad \underset{R}{\operatorname{argmin}} \quad \underset{u\,v \quad R}{\quad} \quad \log \quad _{u\,v} \quad \mathrm{P} \qquad (2.10)$$

can be used if the link descriptor is a “power consumption-like” measure.

Based on this lemma, seeking an MLP with respect to bandwidth requirement becomes a relatively easy task.

DOI:10.15774/PPKE.ITK.2019.006

2.3.2 Novel algorithms for ARII

If the link descriptor is delay, then QoS routing yields an intractable problem stated by the following lemma:

Lemma 2 (Guerin at al.)**.** *The find a solution to ARII*

$$\tilde{R} \quad \underset{R \quad s \quad d}{\operatorname{argmax}} \quad \sum_{u\,v \quad R} d_{u\,v} \quad \mathrm{T} \qquad \text{(2.3 revisited)}$$

in general is NP hard.

The proof is based on the fact that the problem of finding a path which fulfills $\sum_{u\,v \quad R} d_{u\,v} \quad \mathrm{T}$ (where $0 \quad 1$ is some given threshold) is NP hard. For further details, see [58].

One way to make ARII tractable is to reduce it to the bandwidth problem. This gives rise to "rate based" routing algorithms [58]. In practice, under certain scheduling scenarios (such as Weighted Fair Queuing Scheduler, Rate Controlled Earliest Deadline First schedulers), end-to-end delay on an n hop route R can be approximated as

$$d\,R \quad \frac{c\,n}{r} \quad \sum_{u\,v \quad R} d_{u\,v} \qquad (2.11)$$

Where $$ is the size of the flow's burst, c is the maximum packet length for the flow , whereas $d_{u\,v}$ is a static link propagation delay and r is the minimal rate that can be guaranteed to the flow at each link along the path. Thus, in this case ARII reduces to the following problem of finding $\tilde{R}$: $\operatorname{argmax}_R \quad d\,R \quad \mathrm{T}$ [58].

Unfortunately, this problem is still NP hard as shown in [58]. But assuming that $d_{u\,v}$ can take their values from a discrete set and the link descriptor is the available bandwidth and by setting $d \quad {}_{u\,v \quad R} d_{u\,v}$, the problem can still be solved by SPR, as was shown in [58]. In this way, rate based routing can still be performed in polynomial time.

Unfortunately, rate based bounds are rather crude, thus the method above yields only sub-optimal solutions.

In this section I demonstrate that ARII can be solved in polynomial time under certain assumptions. Let us suppose that links in each time instant (this time instant is set up by the network operator) advertise their delay to the nodes, in the following fashion:

- The delay axis (the set of possible values of the link delays) is covered with a grid $t_i \; i \quad 1 \quad Z$.
- At each time instant link $u\,v$ advertises the last t_i value its link delay have exceeded, which implies that ${}_{u\,v} \quad t_i \; t_{i \quad 1}$.

On these premises I derive new algorithms which can find an MLP in polynomial time.

Solving ARII under Gaussian hypothesis Based on the description above, I assume that the concrete delay is unknown in the interval ${}_{u\,v} \quad t_i \; t_{i \quad 1}$, after advertising that the link delay

DOI:10.15774/PPKE.ITK.2019.006

has surpassed level t_i. This prompts us to regard $_{u\,v}$ as a Gaussian random variable, which has normal distribution over the interval $t_i\ t_i\ _1$. One must note, that assuming normal delay distribution is not restrictive, as it is at the modeler's liberty to choose any PDF which lends itself to analytic tractability. Despite the fact that a Gaussian distribution is defined over an infinitely long interval, one can fit a Gaussian distribution $m\ \tilde{}$ over a finite interval with an error by solving the following approximation task:

$$\tilde{} \quad \text{Solve} \int_{t_i}^{t_i\ 1} \frac{1}{\sqrt{2}} e^{\frac{x\ m}{2\ 2}^2} dx \quad 1 \qquad \text{where } m \quad t_i \quad \frac{t_i\ 1 \quad t_i}{2} \quad \frac{t_i\ 1 \quad t_i}{2} \tag{2.12}$$

Based on the normal assumption the following theorem can be proved.

Theorem 1. *If $_{u\,v}$ is a subject to a normal distribution with parameters $\tilde{}_{u\,v}$ $\overline{m}_{u\,v}$, then the solution of ARII*

$$\tilde{R} \quad \underset{R \quad s \quad d}{\operatorname{argmax}} \quad _{u\,v \quad R}^{\quad u\,v} \quad \mathrm{T} \tag{2.3 revisited}$$

is equivalent to minimizing the objective function

$$\tilde{R} \quad \underset{R}{\operatorname{argmin}} \quad m_{\,u\,v} \tag{2.13}$$
$$_{u\,v \quad R}$$

by using the Bellman-Ford algorithm in polynomial time.

Remark 1. *One must note that the LAS is fully characterized on link $u\ v$ if and one boundary (t_i) of an interval is set numerically. and t_i sets m and , since the Gaussian distribution is symmetric and m and t_i sets $t_i\ _1$. So the LAS can be computed recursively from the numerically given interval boundary.*

Remark 2. *Also this means that in this case the Link Advertisement Scheme (LAS) is a non-uniform one, as the changing variance $_{u\,v}$ $\overline{m}_{u\,v}$ indicates that the larger the the delay becomes (large expected value) the larger the inaccuracy becomes in its reported state. See Figure 3 for an example.*

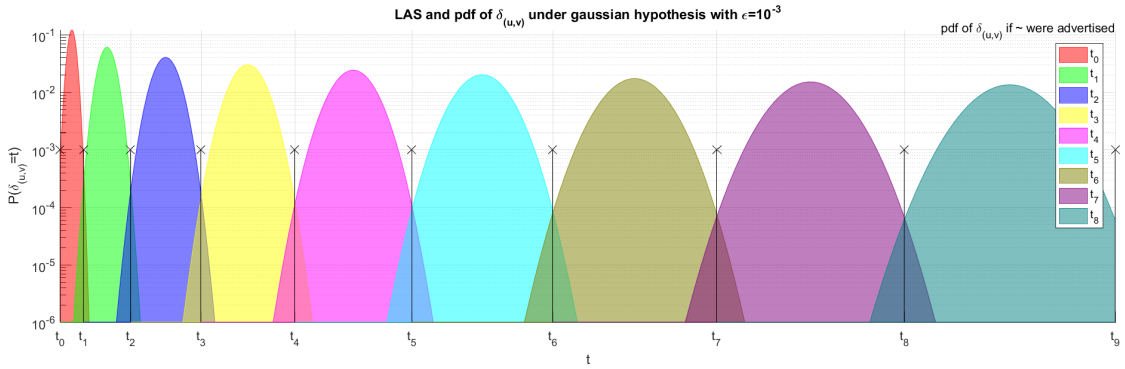


Figure 3: An example LAS with Gaussian approximation

Proof. In case of Gaussian link descriptors the aggregated link descriptor, $q\ R$ $_{u\,v \quad R \quad u\,v}$

DOI:10.15774/PPKE.ITK.2019.006

is also Gaussian (due to the additive nature of delays), thus

$$q\ R \qquad m_{u\,v} \qquad \overline{\qquad\qquad\qquad\qquad} \qquad (2.14)$$
$$u\,v\ \ R \qquad\qquad u\,v\ \ R \qquad {}^{2}_{u\,v}$$

implying that

$$q\ R \qquad \mathrm{T} \qquad \frac{\mathrm{T} \qquad m_{u\,v}}{\overline{\qquad\qquad\qquad\qquad}} \qquad (2.15)$$
$$u\,v\ \ R \qquad {}^{2}_{u\,v}$$

Because of the relationship $_{u\,v}$ $\overline{m_{u\,v}}$ equation can be rewritten as

$$\tilde{R} \quad \operatorname*{argmax}_{R} \quad q\ R \qquad \mathrm{T} \qquad (2.16)$$

$$\operatorname*{argmax}_{R} \quad \frac{\mathrm{T} \qquad {}_{u\,v\ \ R} m_{u\,v}}{\overline{\qquad\qquad\qquad\qquad}} \qquad (2.17)$$
$$u\,v\ \ R\, m_{u\,v}$$

By introducing a new variable M : $m_{u\,v}$ we obtain
$u\,v\ \ R$

$$\tilde{R} \quad \operatorname*{argmax}_{R} \quad q\ R \qquad \mathrm{T} \qquad \operatorname*{argmax}_{R} \quad \frac{\mathrm{T} \quad M}{\overline{M}} \qquad (2.18)$$

Now it is easy to point out that if T 0 and 0, $\frac{\mathrm{T}\ M}{\overline{M}}$ is a monotone decreasing function of M as

$$\frac{d \quad \frac{T\ M}{\overline{M}}}{dM} \qquad \frac{T \quad M}{\overline{M}} \quad \frac{\frac{T}{\overline{M}} \qquad \overline{M}}{2 \quad M} \qquad 0$$

where x : $\frac{1}{\overline{2}} e^{\frac{1}{2}x^2}$. Therefore, minimizing M will force $\frac{T\ M}{\overline{M}}$ to be as large as possible, which indeed leads to the objective function R_{opt} : $\min_R$ $_{u\,v\ \ R} m_{u\,v}$. This objective function is additive, thus the Bellman-Ford algorithm can select optimal path in polynomial complexity. □

THESIS I.1 (unicast routing with incomplete information by Gaussian approximation)**.** *In Theorem 1, I gave a mapping for the link descriptors under the condition that the link descriptors have normal distributions with parameters m and* $\tilde{}_{u\,v}$ $\overline{m_{u\,v}}$ *and also the LAS follows* m $\frac{t_i \ 1\ t_i}{2}$. *Using these assumptions the ARII problem can be reduced to a deterministic traditional SPR.*

The thesis is restated in a self consistent way in Appendix A at Thesis I.1 (page 84).

Finding optimal paths by the Chernoff inequality In this section, I develop QoS routing algorithms by using well-known statistical inequalities from large deviation theory to estimate the tail of the aggregated delay of a path. Our objective can be reformulated to be more suitable for

DOI:10.15774/PPKE.ITK.2019.006

the following discussion. It can be seen that the original problem

$$\tilde{R} \quad \underset{R \quad s \ d}{\operatorname{argmax}} \quad \underset{u\,v \quad R}{\quad} \quad {}_{u\,v} \quad \mathrm{T} \tag{2.3 revisited}$$

is equivalent to:

$$\tilde{R} \quad \underset{R}{\operatorname{argmin}} \quad \underset{u\,v \quad R}{\quad} \quad {}_{u\,v} \quad \mathrm{T} \tag{2.19}$$

Theorem 2. *Using the logarithm of the moment generating function (log-moment generating function)*

$$_{u\,v} \ s \quad \ln \quad \exp \ s \ _{u\,v} \quad \ln \quad \exp \ sx \ dF_{u\,v} \ x \tag{2.20}$$

or in case of a discrete random variable

$$_{u\,v} \ s \quad \ln \quad \exp \ s \ _{u\,v} \quad \ln \underset{i \ 1}{\quad} \exp \ sx_i \ p_i \tag{2.21}$$

the solution of the ARII is equivalent with minimizing the objective function

$$\tilde{R} \quad \underset{R}{\operatorname{argmin}} \quad \underset{u\,v \quad R}{\quad} \quad {}_{u\,v} \quad \hat{s} \tag{2.22}$$

where the optimal $\hat{s}$ parameter is

$$\hat{s} \quad \underset{s}{\inf} \quad \underset{u\,v \quad \tilde{R}}{\quad} \quad {}_{u\,v} \quad s \quad sT \tag{2.23}$$

Proof. The probability $_{u\,v \quad R \quad u\,v} \quad T$ can be upper-bounded by the Chernoff inequality

$$\underset{u\,v \quad R}{\quad} \quad {}_{u\,v} \quad T \quad \exp \quad {}_R \ s \quad sT \tag{2.24}$$

where $_R \ s$ is the log-moment generating function of the aggregated delay, given as $_R \ s$ $_{u\,v \quad R \quad u\,v} \ s$. Therefore instead of minimizing the original quantity I minimize the upper-bound,

$$\tilde{R} \quad \underset{R}{\operatorname{argmin}} \exp \quad \underset{u\,v \quad R}{\quad} \quad {}_{u\,v} \quad s \quad sT \tag{2.25}$$

which yields

$$\tilde{R} \quad \underset{R}{\operatorname{argmin}} \quad \underset{u\,v \quad R}{\quad} \quad {}_{u\,v} \quad s \tag{2.26}$$

This last problem is an SPR with metric $_{u\,v} \quad _{u\,v} \ s$. □

I name the method based on the Chernoff inequality with a given “s” as “Simple Chernoff Algorithm”. One must note that a bound is minimized instead of the true objective function. Thus,

the path found by this method can only be asymptotically optimal. More precisely, since

$$_{u\ v\ \ R}\quad {}_{u\ v}\quad T \quad 1 \quad \exp \quad {}_{R}\ s \quad sT \tag{2.27}$$

one can state that the QoS requirement is met at least with probability 1 where exp $_R$ s sT . Thus this method can quantify the likelihood of satisfying the given QoS parameters, therefore may still prove to be useful from engineering point of view.

The other problem regarding the method is to find the proper value of s which yields the tightest bound. As was seen $\hat{s}$ depends on the path itself. Therefore, choosing an arbitrary s_1 and carrying out the corresponding BF algorithm, it may yield a false result (with another s_2 a totally different path can be found which might yield a better result). This gives rise to Algorithm 1, the "exhaustive-s" algorithm.

Remark 3. *Note that for all practical examples* $_{u\ v}$ s sT *has a minimum and can be found via a simple gradient method. A proof can be found for any finite support discrete variable at Appendix D. At the same time with similar reasoning it can be extended to most of the common continuous random variables.*

Algorithm 1 Exhaustive-s algorithm

Input: $G\ V\ E \quad {}_{u\ v}\ F_{u\ v}\ x \quad src\ dst$

Define a grid on the set of possible values of s denoted by $s_i\ s_i \quad 0\ i \quad 1 \quad M$.

for all i 1 M **do**

Pick s_i .

Perform path selection R_i by an SPR algorithm with link measures

$_{u\ v}\ s_i$: ln exp s_i $_{u\ v}$.

Based on the selected path R_i determine

$$\hat{s}_i \quad \text{Solve} \quad {}_{u\ v\ \ \tilde{R}_i} \frac{d \quad {}_{u\ v} \quad s}{ds} \quad T\ s \tag{2.28}$$

and calculate the bound

$$B_i : \quad \exp \quad {}_{u\ v\ \ R_i} \quad {}_{u\ v}\ \hat{s}_i \quad \hat{s}_i T \tag{2.29}$$

end for

Find the path which belongs to minimal bound

$$\tilde{R}_j : j \quad \operatorname*{argmin}_i B_i \tag{2.30}$$

Output: $\tilde{R}_j$ chosen path between src and dst

It is obvious that the complexity of this algorithm is O $M\ V^2$ which can be overwhelming if M is large. Furthermore, since parameter s can take any positive value but grid is finite, the best path may have been missed by simply ignoring some of the s parameters not being contained by grid . The numerical complexity of the algorithm can be relaxed by making the assumption that in each LAS interval the link descriptor behaves the same, for all $u\ v$ link, which is to say that

DOI:10.15774/PPKE.ITK.2019.006

the delay follows the same PMF over interval $t_i\ t_{i\ 1}$ for any i and any $u\ v$. This assumption is of course quite stringent and introduces errors, but it in enables us to execute the computation fast. For this the original link descriptor $_{u\ v}$ is transformed and discretized into $_{u\ v}$ by splitting every LAS interval into the same number and relatively same length internal intervals.

Figure 4 shows this process. On the top left the original distribution of $_{u\ v}$ is depicted. On the bottom left the conditional distributions are depicted when it is known which LAS interval is active. On the top right the individual conditional distribution pieces are scaled to the interval $0\ 1$. This "generalized" interval will be split into K internal segments. On the picture for the sake of clarity $K\quad 4$, but the larger K the approximation of the underlying distribution becomes more precise. The "discretized" distribution (depicted with green stairs) is taken over the intervals which matches the general behavior of all original conditional distributions. will be the basis of the link independent log-moment generating function. For the sake of clarity on the bottom right picture, is mapped back to the original conditional variable intervals. Note that this example is depicted for a continuous random variable, but it can also be applied to a discrete one.

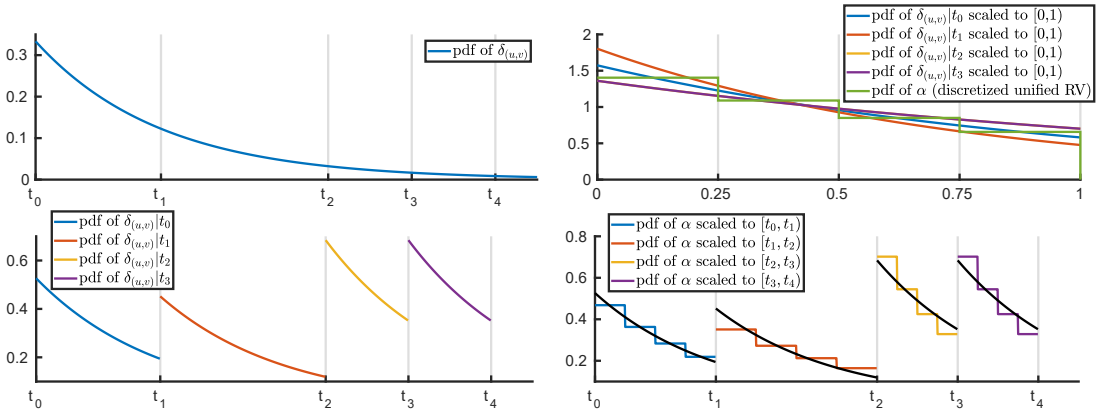


Figure 4: Discretization of the link descriptor

Let us write the conditional pmf of in a slightly different way:

$$_{u\ v}\quad t_i\quad j\quad _i\ t_i \text{ was advertised}\quad P_{i\ j\ \ i}$$
$$_i\quad t_{i\ 1}\quad t_i;\qquad j\quad 0\ \frac{1}{i}\ \frac{2}{i}\qquad 1\ \frac{1}{i};\qquad t_i\quad 1\quad _i\quad t_{i\ 1} \tag{2.31}$$

Let us offset and squeeze every $_{u\ v}\ t_i$ was advertised into the $0\ 1$ interval:

$$_{u\ v\ i}\quad j\qquad _{u\ v}\quad t_i\quad j\quad _i\ t_i \text{ was advertised} \tag{2.32}$$

The assumption I made is that $_{u\ v\ i}$ depends neither on i (the interval advertised) nor on $u\ v$ (the link on which it was advertised). If the assumption would hold and was known, $_{u\ v}\quad t_i$ could be used as the new metric. In general $_{u\ v\ i}$ does depend on i and $u\ v$, but by introducing a discretized and generalized version of it one one hand one can utilize the benefits of it, on the other hand it will introduce another source of error.

I create this generalized from $_{u\ v\ i}$ by splitting up the domain $0\ 1$ into K equal small

DOI:10.15774/PPKE.ITK.2019.006

intervals: $\frac{k}{K} \; \frac{k \; 1}{K} \quad k \quad 0 \qquad K \quad 1$. Furthermore the probabilities for the generalized are needed.

$$k \qquad P_k \quad k \quad 0 \qquad K \quad 1 \tag{2.33}$$

P_k could be take as an appropriate value from the ensemble $\;_{u\,v\;\;i}$ for the small interval indexed by k. For example the mean value of the event:

$$P_k \qquad \underset{u\,v\;\;i}{\text{mean}} \qquad _{u\,v\;\;i} \text{ is in } \frac{k}{K} \; \frac{k \quad 1}{K} \qquad C \tag{2.34}$$

where C is a constant that ensures that $\;_k P_k \quad 1$. Note that the simplest form of this discretization is to choose $K \quad 1$, so every LAS interval becomes a "flat" distribution with $P_0 \quad 1$ for every interval. Using the PMF of the link descriptor in any LAS interval is given by:

$$_{u\,v} \qquad a_{\,u\,v} \qquad\qquad P_k \tag{2.35}$$

where k is used to index the small interval within the LAS interval $\; t_i \; t_i \;\; _1$ and the currently advertised felt boundary of the interval is denoted by $a_{\,u\,v} \qquad t_i$ Then $\;_{u\,v}$ is given as

$$_{u\,v} \;\; s \qquad \ln \sum_{k\;\;0}^{K\;\;1} P_k \exp \;\; s \;\; a_{\,u\,v} \qquad k \tag{2.36}$$

which yields

$$_{u\,v} \;\; s \qquad s \;\; a_{\,u\,v} \qquad\quad s \tag{2.37}$$

where

$$s \;\; : \quad \ln \sum_{k\;\;0}^{K\;\;1} P_k \exp \;\; s \;\; k \tag{2.38}$$

is a link independent general log-moment generator function. Based on this assumption, the following lemma can be stated:

Lemma 3. *For the QoS of a given path R the sharpest bound can be obtained as:*

$$\sum_{u\,v \quad R} {}_{u\,v} \qquad T \qquad \exp \sum_{u\,v \quad R} {}_{u\,v} \;\; \tilde{s} \qquad \tilde{s}T \tag{2.39}$$

where

$$\tilde{s} \qquad \frac{{}^{1} \quad T \qquad \sum_{u\,v \quad R} a_{\,u\,v}}{R} \tag{2.40}$$

In the expression above $\;{}^{1} \; x \;$ is the inverse of the derivative of the link independent log-moment generating function $\; s \;$, which always exist since I am modeling non negative link descriptors (see D) and $\; R \;$ indicates the number of hops in path R.

Proof. To obtain the sharpest bound, s has to be optimized as follows:

$$\tilde{s} : \inf_{s} \sum_{u\,v \quad R} {}_{u\,v} \;\; s \qquad sT$$

DOI:10.15774/PPKE.ITK.2019.006

This can be achieved by differentiation (see [102, 78], D), yielding

$$\tilde{s}: \inf_{s} \quad \frac{d \quad _{u\,v} \quad s}{ds} \quad T$$

Taking into account that $_{u\,v} \quad s \quad sa_{u\,v} \quad s$ and performing the differentiation, one can obtain

$$\tilde{s} \quad T \quad a_{u\,v}$$

which indeed yields

$$\tilde{s} \quad \frac{^{1} \; T \quad _{u\,v} \quad R a_{u\,v}}{R}$$

□

Based on Lemma 3 one can easily calculate the $\tilde{s}$ which belongs to path R. To illustrate this dependence, I will use the notation $\tilde{s}\ R$ in the forthcoming discussion.

Due to the optimization of s according to Lemma 3, another method can be proposed to find an MLP, which I will refer to as "Recursive Path Finder - s Finder Algorithm". The name originates from the fact that with a given s first it finds an optimal path $R\ s$ then for this path it determines $\tilde{s}\ R$ and searches for a new path with that updated s parameter, ...etc.

Algorithm 2 The Recursive Path Finder - s Finder Algorithm

Input: $G\ V\ E \quad _{u\,v} \quad F_{u\,v} \quad x \quad src\ dst$
Pick $s \quad$ a positive starting value
compute the path independent $\quad s$
repeat
Associate measure $\quad _{u\,v} \quad s$ to each link $u\ v \quad E$.
Perform the SPR algorithm to find the optimal path $\tilde{R}\ s$ for parameter s.
For the obtained $\tilde{R}$ determine $\tilde{s}$ by expression

$$\tilde{s} \quad \frac{^{1} \quad T \quad _{u\,v} \quad R \quad a_{u\,v}}{R} \qquad \text{(2.40 revisited)}$$

$s \quad \tilde{s}$.
until $\tilde{R}\ \tilde{s} \quad \tilde{R}\ s$
Output: $\tilde{R}\ s$ chosen path between src and dst

From Algorithm 2 it is clear that in each step either the quality of the path is improved (for a given s it finds an optimal $\tilde{R}$) or the quality of the bound is improved (for a given path it finds an optimal $\tilde{s}$). Therefore, by carrying out the algorithm recursively, the solution is always improved. The algorithm will settle in a fix point if the following equation holds :

$$\tilde{R}\ \tilde{s} \quad \tilde{R}\ s \qquad (2.41)$$

meaning that for a given s the optimal path $\tilde{R}$ remains the same as the path provided by the BF

DOI:10.15774/PPKE.ITK.2019.006

algorithm which is run again with the optimized $\tilde{s}$ $\tilde{R}$. (The convergence of this algorithm can be proved by demonstrating that R s : $_{uv}$ R $_{uv}$ s sT is a Lyapunov function associated with the "Recursive Path Finder - s Finder Algorithm".)

The method given above relies on the fact that the optimal $\tilde{s}$ can be calculated for a given path R relatively simply. This simplicity can be maintained if the log-moment generating function of link delays are calculated as $_{uv}$ s sa_{uv} s ,where s is the link independent part (see (2.38)).

THESIS I.2 (unicast routing with incomplete information by recursive path finder algorithm)**.** *In Algorithm 1 and Algorithm 2, I gave procedures that can find routes in a packet switched network which satisfy the required QoS parameter with a given probability . The algorithms are based on a transformation of the random link descriptors using the large deviation theory which is described in Theorem 2.*

The thesis is restated in a self consistent way in Appendix A at Thesis I.2 (page 84).

2.4 Multicast routing in WSN applications with incomplete information

The main contribution of this section is the method to transform the original probabilistic link descriptors into deterministic measures, which reduces the multicast routing with incomplete information into a GSMT search. Then I apply the HNN which ensures fast convergence to a sub-optimal solution[161].

To investigate the problem with realistic quantities, a multicast scenario is used on a WSN where the QoS metrics are of the same bandwidth or delay type but with the extension that we need to deal with the power consumption of the individual nodes. To prolong node lifetime we need to minimize the radio transmission. This specific scenario can be generalized to other types of applications as well. I use a simple model for the radio communication between the nodes in

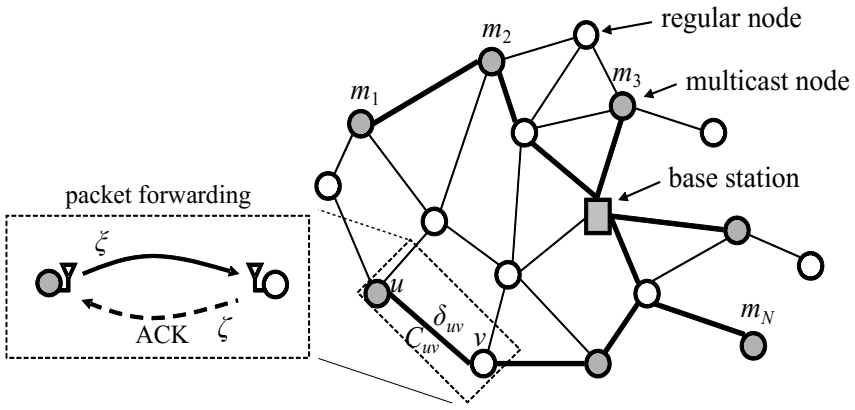


Figure 5: A typical Wireless Sensor Network, with Base Station denoted as a gray box, multicast nodes as gray circles and regular nodes as white circles.

the network. It is assumed that equal transmission power is used by each node

$$\mathrm{P}_{uv}^{\mathrm{TX}} \quad g \qquad u \; v \quad E \tag{2.42}$$

DOI:10.15774/PPKE.ITK.2019.006

The probability of successful packet reception is calculated by using the Rayleigh fading model [60] as follows

$$\text{packet reception by } u \text{ from } v \quad \mathrm{p}_{uv} \quad \exp \quad \frac{\frac{2}{z}\, d_{uv}}{g} \tag{2.43}$$

where is a hardware related threshold, $\frac{2}{z}$ represents the noise power, d_{uv} denotes the distance between node u and v, and is the path loss exponent.

A communication scheme is assumed where receiver nodes use ACKs, and it is assumed that the number of ACKs are not limited. Table 2 shows a sequence of packet transmissions between source node u and destination v, where v can either receive or fail to receive the data packet, which cases are denoted by 1 and 0 respectively. The ACK can also be decoded correctly by u or not. The communication ends when both packets are received correctly. Let us denote the event of

Table 2 A sequence of packet reception until both data and ACK are received.

Events	Outcomes					
Reception of data from u to v	1	0	1	...	1	1
Reception of ACK from v to u	0	0	0	...	0	1

successful data reception followed by no ACK reception – first column – with and the event of unsuccessful data reception – second column – with . The corresponding distributions can be expressed as follows

$$_{uv} \quad m \quad 1 \quad \mathrm{p}_{vu}{}^{m} \quad \mathrm{p}_{vu} \quad \text{and} \tag{2.44}$$

$$_{uv} \quad n \quad \frac{\mathrm{p}_{uv} \quad \mathrm{p}_{vu} \quad 1 \quad \mathrm{p}_{uv}{}^{n}}{1 \quad \mathrm{p}_{uv} \quad \mathrm{p}_{uv} \quad \mathrm{p}_{vu}{}^{n \;\; 1}} \tag{2.45}$$

Let us define as the random variable corresponding to the power consumption over the link $u \;\; v$ until successful data transmission

$$_{uv} \quad _{uv} \; 2g \quad _{uv} \; g \quad 2g \tag{2.46}$$

C_{uv} the expected value of the transmit power over link $u \;\; v$ then is

$$C_{uv} : \quad _{uv} \quad _{uv} \quad \mathrm{P}_{uv}^{\mathrm{TX}} \quad \mathrm{P}_{vu}^{\mathrm{TX}} \quad _{uv} \quad \mathrm{P}_{uv}^{\mathrm{TX}} \quad \mathrm{P}_{uv}^{\mathrm{TX}} \quad \mathrm{P}_{vu}^{\mathrm{TX}}$$
$$_{uv} \quad 2g \quad _{uv} \quad g \quad 2g \tag{2.47}$$

In this case the distribution of the delay $_{u\,v}$ on link $u \;\; v$ can be calculated as follows:

$$_{uv} \quad l \quad \mathrm{p}_{uv} \;\; \mathrm{p}_{vu} \quad 1 \quad \mathrm{p}_{uv} \;\; \mathrm{p}_{vu}{}^{l \;\; 1} \quad l \quad 1 \;\; 2 \tag{2.48}$$

Due to incomplete information about the link states in the network, link metrics $_{u\,v}$ are described by random variables. These link metrics are not additive thus the deterministic multicast tree search algorithms are not applicable. We transform the random link metrics into deterministic descriptors which can be fed to the traditional or heuristic algorithms. In order to do this, I

DOI:10.15774/PPKE.ITK.2019.006

introduce two type of common requirements for designing a multicast tree: a bottleneck type and an end-to-end additive type requirement.

The deterministic bottleneck Steiner tree problem is formulated (see subsection E.1) as

$$\max_{u\,v\quad \mathrm{A}} \mathrm{P}^{\mathrm{TX}}_{uv} \qquad \mathrm{A} \tag{2.49}$$

where is the set of all trees containing src M, $\mathrm{P}^{\mathrm{TX}}_{uv}$ is the transmitter power consumed during transmission, and is the limit that one does not want to exceed in order to economize battery power.

The deterministic end-to-end additive type problem (see subsection E.2) can be formulated as

$$\begin{aligned} &\operatorname*{argmin}_{\mathrm{A}} \qquad C_{uv} \\ &\text{s.t.} \qquad \sum_{u\,v\quad R_{src\,m}} D_{uv} \quad \mathrm{T} \qquad m \quad M \end{aligned} \tag{2.50}$$

where $R_{src\,m}$ is a path from src to m M in the multicast tree A, and D_{uv} is an additive metric (like the delay) with QoS requirement T and C_{uv} is the expected used power.

Although no polynomial algorithm is known for finding Steiner trees but sub-optimal algorithms exist[138]. In the next subsections I extend the problem to the case of random link descriptors instead of the deterministic metrics and propose a heuristic method using a HNN.

2.4.1 Bottleneck type requirement

Similarly as in the unicast case (see subsubsection 2.3.1) a lemma can be formulated for the bottleneck case. In this case the objective is to find an optimal tree $\tilde{\mathrm{A}}$ from all multicast trees $^{src\quad M}$ which most likely fulfills the given QoS criterion.

Lemma 4. *Assuming independent random bottleneck type link descriptors* $_{u\,v}$ $_{uv}$*, the optimal multicast tree problem*

$$\tilde{\mathrm{A}}_1 \quad \operatorname*{argmax}_{\mathrm{A}\quad src\quad M} \quad \max_{u\,v\quad \mathrm{A}} \quad _{u\,v} \quad \mathrm{P} \qquad \text{(2.4 revisited)}$$

is equivalent of a GSMT problem with metric

$$\tilde{\mathrm{A}}_1 \quad \operatorname*{argmin}_{\mathrm{A}\quad src\quad M} \quad \sum_{u\,v\quad \mathrm{A}} \ln \quad _{uv} \quad \mathrm{P} \tag{2.51}$$

Proof. If $_{uv}$ P holds for the maximal $_{uv}$ then it holds for all the values in the tree, which yields

$$\max_{u\,v\quad \mathrm{A}} \quad _{uv} \quad \mathrm{P} \tag{2.52a}$$

$$\sum_{u\,v\quad \mathrm{A}} \quad _{uv} \quad \mathrm{P} \tag{2.52b}$$

DOI:10.15774/PPKE.ITK.2019.006

$$\sum_{u\,v \quad \mathrm{A}} \quad {}_{uv} \quad \mathrm{P} \tag{2.52c}$$

$$\sum_{u\,v \quad \mathrm{A}} \ln \quad {}_{uv} \quad \mathrm{P} \tag{2.52d}$$

where I used the independence property of these random variables.

This results in the following objective function over additive measures:

$$\tilde{\mathrm{A}}_1 \quad \underset{\mathrm{A} \quad {}^{src} \quad M}{\operatorname{argmin}} \sum_{u\,v \quad \mathrm{A}} \ln \quad {}_{uv} \quad \mathrm{P} \tag{2.51 revisited}$$

□

which is well suited for the later introduced HNN.

2.4.2 End-to-end additive requirement

In the traditional multicast setting with incomplete information the goal is to search for a tree with maximal probability on which the delay ${}_{uv}$ of the longest route is being smaller than T

$$\underset{\mathrm{A}}{\operatorname{argmax}} \quad \max_{R_{src\,m} \quad {}_{\mathrm{A}}} \quad \prod_{u\,v \quad R_{src\,m}} {}_{uv} \quad \mathrm{T} \tag{2.53}$$

Where ${}_{\mathrm{A}}$ denotes all the possible routes in the tree. The WSN setting requires us to also minimize the transmit power C_{uv}(see (2.47)) used by the network in order to prolong node life span. To incorporate this additional constraint I define the following optimization problem, where the constraint expresses the probability that the "worst" route in the chosen tree does not satisfy the constraint.

$$\begin{aligned} \tilde{\mathrm{A}}_2 : & \underset{\mathrm{A}}{\operatorname{argmin}} \sum_{u\,v \quad \mathrm{A}} C_{uv} \\ \text{s.t.} \quad & \max_{R_{src\,m} \quad {}_{\mathrm{A}}} \quad \prod_{u\,v \quad R_{src\,m}} {}_{uv} \quad \mathrm{T} \end{aligned} \tag{2.54}$$

In the optimal Steiner tree there are common links between paths for different multicast nodes, meaning that the random measures for different paths are statistically dependent, which can be described by the joint distribution function.

We can use large deviation theory to approximate the previous probability

$$\max_{R_{src\,m} \quad {}_{\mathrm{A}}} \quad \prod_{u\,v \quad R_{src\,m}} {}_{uv} \quad \mathrm{T} \quad \exp \quad {}_{R_{src\,m}} \; s \quad s \; \mathrm{T} \tag{2.55}$$

where

$${}_{R_{src\,m}} \; s \quad \prod_{u\,v \quad R_{src\,m}} {}_{uv} \; s \quad \text{and} \tag{2.56}$$

DOI:10.15774/PPKE.ITK.2019.006

$$_{uv}\ s \quad \ln \quad \exp\ s \quad _{uv} \tag{2.57}$$

The optimal s value can be calculated by solving

$$s_{\text{opt}} : \quad _{u\ v \quad R_{src\ m}} \frac{d}{ds} \ _{uv}\ s \quad \text{T} \tag{2.58}$$

where $R_{src\ m}$ is a given route in the multicast tree from s to m.

We are looking for the tree approximated by Equation (2.55) and our sub-optimal solution will be the one with the minimal $_{u\ v \quad \text{A}} C_{uv}$ for which

$$\exp \quad _{R_{src\ m}}\ s \quad s \ \text{T} \tag{2.59}$$

$$_{R_{src\ m}}\ s \quad \ln \quad s \ \text{T} \tag{2.60}$$

which translates Equation (2.54) into

$$\begin{aligned} \tilde{\text{A}}_2 : &\underset{\text{A}}{\text{argmin}} \quad _{u\ v \quad \text{A}} C_{uv} \\ &\text{s.t.} \quad _{R_{src\ m}}\ s \quad \ln \quad s \ \text{T} \end{aligned} \tag{2.61}$$

which is in the form of the well-known CGSMT [130], for which a DHNN based approximation is given in subsubsection 2.4.3.

For a given T parameter then better and better trees can be found by decreasing in an iterative or gradient fashion, which yields Algorithm 3.

Algorithm 3 Find optimal tree for end-to-end requirement

Input: $G\ V\ E \quad _{u\ v}\ F_{u\ v}\ x \quad , \quad 1, \text{T} \quad 1\ src,m$
  repeat
    A find tree with HNN G T
    if A is found **then**
      decrease
    else
      increase
    end if
  until no significant increase in performance
Output: A is the multicast tree between *src* and *m*

Figure 6 illustrates a case of two sets of trees having equal longest path delay properties – from each set I choose based on the sum of transmit powers –. Assume that it is required that the probability of the longest route exceeding 6 is to be maximized. At the first step (having $_1$) routes in both trees can guarantee a longest delay of 6 with probability $_1$ thus can be decreased. In the second step ($_2$) still both trees satisfy the condition while in the third step there is only one tree below delay 6. This way the minimal value of can be determined for which there exists at least one tree.

DOI:10.15774/PPKE.ITK.2019.006

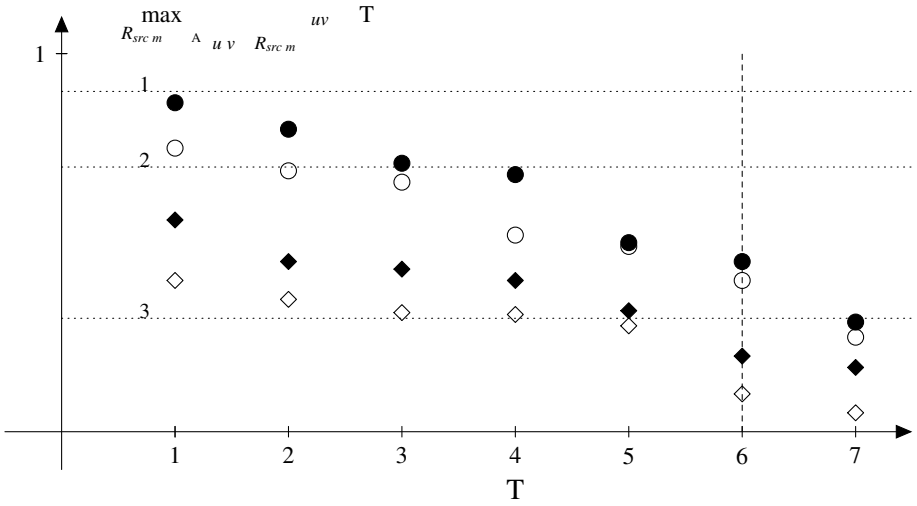


Figure 6: The probability of longest route exceeding threshold T for two trees ♦ and . The approximated Chernoff-bound for these probabilities are ◊ and , respectively.

2.4.3 Approximation by HNN

In this part I derive the necessary structure that can enable us to construct a HNN to approximate the defined problem above. Hopfield Networks in general have successfully been applied to combinatorial optimizations and solved many practical tasks[158, 44, 105, 75] as also mentioned in B. So this problem's solution can be approximated by HNN as well[130, 20, 151], based on the energy function proposed in [130, 20]. We use DHNN, because it is reported to be computationally more effective [40, 41, 39]. The energy function is a weighted linear combination of terms which are describing the objective function to be minimized E_1 and the press of the constraint function subjected by the minimization task E_5^m. The feasibility of the solution is guaranteed by the neuron update selection rule, which ensures transitions to only valid candidate solutions. The HNN searches for routes and for a tree solution the union of the chosen routes are used. Thus it is implicitly assumed that the union of routes, satisfying the constraints is a good Steiner tree. We have N nodes in the graph G, so possibly N^2 edge can exist. Every neuron represents an edge in the graph [130] and one neuron's output is noted by V_{uv} which is defined as

$$V_{uv} \quad \begin{array}{ll} 1 & \text{if the link (u,v) is connected} \\ 0 & \text{otherwise} \end{array} \tag{2.62}$$

Note that I denote with V_{uv}^m those edges that are already chosen for a path from *src* to m.

Cost and constraint terms The cost value for the edge $u\ v$ is noted by C_{uv}. The cost of a particular edge configuration is denoted by $E_1\ G$.

$$E_1\ G \quad \begin{array}{c} N\ \ N \\ u\ \ 1\, v\ \ 1 \\ u\ \ v \end{array} \frac{C_{uv}V_{uv}}{1 \qquad V_{uv}^m} \\ m\ \ D \tag{2.63}$$

DOI:10.15774/PPKE.ITK.2019.006

where D denotes set of the multicast node indices we have already chosen a route for. So $_{m\ D}V_{uv}^{m}$ k if we have chosen edge $u\ v$ for k times in previous routes. Note that because of V_{uv}^{m} the cost term is dependent on the edges previously elected in the multicast tree, edge reuse are preferred. E_2 is the term which presses the constraint function to be true. L_{uv} 0 is the delay value on the link $u\ v$.

$$\sum_{\substack{u\ 1\\ v\ 1\\ v\ u}}^{N\ N} L_{uv}V_{uv} \quad \ln \quad s\ T \tag{2.64}$$

We use the same approach as [130] for dealing with inequality constraints, namely introduce a *linear programming* type neuron with cost function $h\ z$ and the corresponding energy term $H\ z$ $h\ z\ dz$.

$$h\ z \quad \begin{array}{ll} z & \text{if } z \quad 0 \\ 0 & \text{otherwise} \end{array} \qquad H\ z \quad \begin{array}{ll} z^2\ 2 & \text{if } z \quad 0 \\ 0 & \text{otherwise} \end{array} \tag{2.65}$$

$$E_2\ G \quad H \sum_{\substack{u\ 1\\ v\ 1\\ v\ u}}^{N\ N} L_{uv}V_{uv} \quad \ln \quad s\ T \tag{2.66}$$

Neuron selection rule for DHNN We initialize the network, that there is only one edge in the chosen path: V_{uv} 1 if $u\ v$ $src\ m$; 0 otherwise. We update exactly 3 neurons for one discrete time step as follows:
Selection A (edge splitting): choose an edge which is in the path $u\ v$ $R_{src\ m}$, choose two edges which are not in the path, but join at a common node and start and terminate at u and v. $u\ w$ $w\ v$ $R_{src\ m}$.
Selection B (edge joining): choose two edges which are in the path as $u\ w$ $w\ v$ $R_{src\ m}$, thus $u\ v$ $R_{src\ m}$.
Either A or B used, update the three neurons (flip V_{uv} V_{uw} V_{wv} triangle) if the state transition yields to a better energy state of the network. Use A and B alternatively until no state transition occurs. This selection rule ensures that if we started from a valid route we end up in a valid route from *src* to *m*.

Note that our graph models a wireless communication network which in theory is fully connected, but a connection may have bad weight characteristics (near infinite energy or near 0 reception probability). So that edge can be chosen, but the cost terms will rule them out.

Cost and constraint terms for DHNN The conventional energy function of the discrete Hopfield model is

$$_{\text{DHNN}}\ \mathbf{y} \quad \mathbf{y}^{tr}\mathbf{W}\mathbf{y} \quad 2\mathbf{y}^{tr}\mathbf{b} \quad \mathbf{y} \qquad 1 \quad 1\ ^{K} \tag{2.67}$$

So the aim is to transform $_1\ E_1\ G$ $_2\ E_2\ G$ into the previous quadratic form. Where $_1$ and $_2$ are the usual weight factor to emphasize one constraint against the other. Note that to avoid confusion with parameter T -only in this chapter- I denote matrix-vector transposition with $\mathbf{A}^{tr}$. We transform the neurons output variable to a column vector with elements of 1 1 for the

DOI:10.15774/PPKE.ITK.2019.006

DHNN. We have N nodes in the graph ($u\ v \quad 1 \quad N$) and $\mathbf{V}$ is read out column wise.

$$\mathbf{v} : v_k \quad V_{uv} \quad y_k \quad 1 \quad 2 \tag{2.68}$$

$$\mathbf{y} : y_k \quad 2v_k \quad 1 \tag{2.69}$$

$$u \quad \frac{k \quad 1}{N} \quad 1 \quad v \quad k \quad 1 \mod N \quad 1 \tag{2.70}$$

We introduce the abbreviation $^{u\ v}$ for the elimination of the terms "$u \quad v$" from the summation by the multiplication with it.

$$^{u\ v} : \quad {}^{u}_{i}\,{}^{v}_{j} \quad \begin{matrix} 1 & \text{if} \quad i \quad 1 \mod N \quad 1 \quad j \quad 1 \quad N \quad 1 \\ & \text{and } i \quad j \\ 0 & \text{otherwise} \end{matrix} \tag{2.71}$$

$$^{u\ v} \quad \mathbf{I}_{N^2\ N^2} \quad {}^{u\ v} \quad {}^{u\ v} \quad {}^{N^2\ N^2}_{N\ N} \quad V_{uv} \quad \mathbf{1}_{1\ N} \quad {}^{u\ v}_{u\ 1 v\ 1} \quad \mathbf{v} \tag{2.72}$$

For the term $E_1\ G$ after the transformation will be

$$b_k^{1\dagger} \quad \frac{C_{uv}}{1 \quad {}_{m\ D} V_{uv}^{m}} \tag{2.73}$$

$$\mathbf{b}^{\ 1} \quad \frac{1}{2} \quad {}^{u\ v} \quad \mathbf{b}^{1\dagger}$$

$$\mathbf{W}^{\ 1} \quad \mathbf{0} \tag{2.74}$$

Similarly for E_2 the calculation can be done. First we read out L_{uv} column-wise using indices as in (2.70), $\mathbf{l} : l_k \quad L_{uv}$.

$$E_2\ G \quad \frac{1}{2} \quad {}^{N\ N}_{u\ 1 v\ 1}{}_{v\ u} \quad L_{uv} V_{uv} \quad \ln \quad s\ \mathrm{T} \quad {}^{2} \quad \frac{1}{2} \quad \mathbf{l}^{tr} \quad {}^{u\ v} \quad \mathbf{v} \quad \ln \quad s\ \mathrm{T} \quad {}^{2}$$

$$\frac{1}{2} \quad \mathbf{v}^{tr} \quad {}^{u\ v\ tr} \quad \mathbf{l}\ \mathbf{l}^{tr} \quad {}^{u\ v} \quad \mathbf{v} \quad 2 \ln \quad s\ \mathrm{T}\ \mathbf{l}^{tr} \quad {}^{u\ v} \quad \mathbf{v} \quad \ln \quad s\ \mathrm{T}^{\ 2} \tag{2.75}$$

$$\mathbf{b}^{\ 2} \quad \frac{\ln \quad s\ \mathrm{T}\ \mathbf{l}^{tr} \quad {}^{u\ v}}{4} \tag{2.76}$$

$$\mathbf{W}^{\ 2} \quad \frac{{}^{u\ v\ tr}\ \mathbf{l}\ \mathbf{l}^{tr} \quad {}^{u\ v}}{8} \tag{2.77}$$

DOI:10.15774/PPKE.ITK.2019.006

The energy function of the HNN as in (2.67) is

$$\mathbf{y} \quad {}_1 \; 2\mathbf{y}^{tr}\mathbf{b}^{\,1} \qquad {}_2 \; \mathbf{y}^{tr}\mathbf{W}^{\,2}\,\mathbf{y} \quad 2\mathbf{y}^{tr}\mathbf{b}^{\,2} \tag{2.78}$$

where $_1$ and $_2$ are the usual weight factor to emphasize one constraint over the other.

THESIS I.3 (multicast routing with incomplete information with HNN)**.** *I defined an algorithm to find a sub-optimal solution to the multicast routing problem with random link descriptors in Algorithm 3. The procedure transforms the random link descriptors into deterministic ones by using results from large deviation theory, which I formulated at* (2.61)*. The transformed problem can be seen as a CGSMT, which is still NP-hard, but I propose a sub-optimal solution by using HNN, where the corresponding parameters are described at 2.4.3 and summarized in* (2.78)*.*

The thesis is restated in a self consistent way in Appendix A at Thesis I.3 (page 86).

2.5 Optimal link scaling as a constrained optimization problem

The link scaling problem, which was defined at 2.2.3 can be treated as a constrained optimization problem, where the Signaling Bandwidth (SB) (the average code length needed by the link state advertisement) has to be kept under a specified threshold while at the same time the Link Entropy (LE) (which describes the uncertainty of the available information about the network) has to be minimized.

A remark: $_{u\,v}$ is an information theoretical source, which generates the LSA for link $u\ v$. has its alphabet from the LAS elements $(t_0\ t_1 \qquad t_{Z\ 1})$ and emits them with probabilities $\hat{p}_0\ \hat{p}_1 \qquad \hat{p}_{Z\ 1}$. The source is active ($_{u\,v}$ =1) with probability $\hat{P} \qquad {}_{u\,v} \quad 1 \qquad {}_i\hat{p}_i$, and inactive with probability $1 \quad \hat{P} \qquad {}_{u\,v} \quad 0$. The event that $t_i\ t_{i\ 1}$ interval was advertised for this link $(t_i \qquad {}_{u\,v} \quad t_{i\ 1})$, is denoted by $_{u\,v} \quad i$. We can define the following conditional entropy corresponding to an active source:

$$\mathrm{H} \quad {}_{u\,v} \quad {}_{u\,v} \quad 1 \qquad \sum_{\substack{i \;\; 0 \\ i:\hat{p}_i \;\; 0}}^{Z\ 1} \frac{\hat{p}_i}{\hat{P}} \log_2 \quad \frac{\hat{p}_i}{\hat{P}} \tag{2.79}$$

$$\mathrm{H} \quad {}_{u\,v} \quad {}_{u\,v} \quad 0 \quad 0 \tag{2.80}$$

Then the quantity $\mathrm{H} \quad {}_{u\,v} \quad {}_{u\,v} \qquad \hat{P}\ \mathrm{H} \quad {}_{u\,v} \quad {}_{u\,v} \quad 1$ is called the Signaling Entropy (SE). By the source coding theorem it is bound between

$$\mathrm{H} \quad {}_{u\,v} \quad {}_{u\,v} \qquad \mathrm{SE} \quad \mathrm{H} \quad {}_{u\,v} \quad {}_{u\,v} \quad 1 \tag{2.81}$$

This is the achievable lower bound for the Signaling Bandwidth (SB) and can be approximated by Huffman coding.

In order to take into account the routing performance resulting from the incomplete link state information, let us assume that the random link descriptor $_{u\,v}$ can take its value from a discrete set: $x_j\ j \quad 0 \qquad L$. Let us denote with $\tilde{p}_i\ j$ the normalized probability of $_{u\,v} \quad x_j$, given that

DOI:10.15774/PPKE.ITK.2019.006

we know that $_{u\,v}$ is in interval $t_i\ t_{i\ 1} : \tilde{p}_{i\ j}$ $_{u\,v}$ $x_j\ t_i$ $_{u\,v}$ $t_{i\ 1}$ and a set of indices for this interval with $I^{\,i}$, where $j \quad I^{\,i}$ if $t_i \quad x_j \quad t_{i\ 1}$. The conditional uncertainty of the random link descriptor in $t_i\ t_{i\ 1}$ can be calculated as:

$$H_i \quad \mathrm{H} \quad _{u\,v}\ t_i \quad _{u\,v} \quad t_{i\ 1} \quad \mathrm{H} \quad _{u\,v} \quad _{u\,v} \quad i \quad \tilde{p}_{i\ j} \log_2 \tilde{p}_{i\ j} \qquad (2.82)$$

The total uncertainty on link $u\ v$ (LE) is defined as:

$$\mathrm{H} \quad _{u\,v} \quad _{u\,v} \quad \int_{i\ 0}^{Z\ 1} \quad _{u\,v} \quad i \quad H_i \qquad (2.83)$$

where $_{u\,v} \quad i \quad t_i \quad _{u\,v} \quad t_{i\ 1}$. Note that both $_{u\,v}$, $_{u\,v}$ depend on the choice of the LAS, consequently the LE and SB also depend on it. The task to solve is the following: how to choose an appropriate LAS that minimizes the LE and keeps the SB under a given threshold. For the forthcoming discussion -not to overwhelm the reader with another complex notation- I assume an equidistant LAS with interval size t, but the same argument can be used for any LAS.

2.5.1 Traffic modeling and queuing system

To be able to compute these metrics, I model the incoming traffic on a link as a Markovian Arrival Process (MAP) process. The MAP process has been successfully applied in network traffic modeling [120], since it has a rich internal structure and can capture the short term correlation in the data, which enables the model to behave more realistically (e.g. generate bursty traffic as well). I choose to model the router and the traffic as a MAP/M/1 queuing system, which is a special case of a more general Quasi Birth-Death (QBD) queue [123, 122]. Queuing systems and their stationer behavior are extensively analyzed for many types of queuing systems. (e.g. M/G/1[154], GI/M/1[122], GI/G/1[43], QBD[122, 124, 123], MAP/PH/1[123], MMPP/M/1[37, 67, 97]). In this section I only introduce the bare minimum from the common notations, definitions and results for the purpose of understanding the next section. I am applying the results to arrive at the computability of information theoretic metrics I described previously. The reader is referred to the articles mentioned before for details about these structures and methods.

The MAP process is a point process governed by an underlying continuous time Markov chain. This Markov chain has a M states. The process can stay in a state $i \quad i$ or transition to another state $i \quad j\ i \quad j$. If the process stays in a state it happens with an arrival of a packet. If the process makes a state transition, it can have it in two fundamental ways: with or without an arrival of a packet. The MAP is described by two matrices MAP **D0** **D1**. **D0** describes the rates of state transitions without arrivals (type 0 transitions) and **D1** describes the rates with arrivals (type 1 transitions). The underlying Markov chain has the generator **D** **D0** **D1**. For **D** to be a generator, the diagonal elements of **D0** are defined as $D0_{ii} \quad _{j\ i} D0_{ij} \quad _j D1_{ij}$. Consider a simpler special case for a MAP, the MMPP: for the MMPP **D1** is diagonal, meaning that when an arrival happens the chain does not change state, but on the other hand when the chain changes state, it won't be associated with an arrival. For example in case of MMPP(3) the chain has 3 states. Each state is associated with a Poisson arrival process with rate $_1 \quad _2 \quad _3$. If the chain

DOI:10.15774/PPKE.ITK.2019.006

is in state i packets arrive with rate $_i$ (The inter arrival times are exponentially distributed with mean 1 $_i$) And the system can transition from i j i j with rates $_{ij}$ described by **D0**.

$$\mathbf{D0} \quad \begin{matrix} 1 & 12 & 13 & & & & & & \\ & 21 & & 2 & 21 & 23 & & & \\ & 31 & & & 32 & & 3 & 31 & 32 \end{matrix} \qquad \mathbf{D1} \quad \begin{matrix} 1 & & \\ & 2 & \\ & & 3 \end{matrix} \tag{2.84}$$

A general QBD queue is an FCFS queue, where the underlying operation is governed by yet another continuous time Markov chain. This Markov chain has a slightly complicated state space, i 0 1 i n . The state space can be divided to levels, denoted by , which correspond to the number of elements in the queue, and to phases, denoted by i. At level the process has n phases. A phase describes an "operation mode" of the chain. In a QBD, transitions are only allowed to either a neighboring or the same level i 1 k j 1 m j i, which correspond to a departure of a packet (with or without a phase change), a pure phase change, or a packet arrival (with or without a phase change) respectively. The infinitesimal generator matrix (transition rate matrix) of a general QBD queue has a tridiagonal block structure (using the notation from [123]):

$$\mathbf{Q} \quad \begin{matrix} \mathbf{L}^0 & \mathbf{F}^0 & & & \\ \mathbf{B}^1 & \mathbf{L}^1 & \mathbf{F}^1 & & \\ & \mathbf{B}^2 & \mathbf{L}^2 & \mathbf{F}^2 & \\ & & \ddots & \ddots & \ddots \end{matrix} \tag{2.85}$$

where sub-matrices **B** encode the backward transitions from level to 1, **F** encode the forward transitions from level to 1 and **L** encode the local transitions within level .

Consider the chosen MAP/M/1 queue, where packets arrive according to MAP **D0 D1** , the service of a packet is according to a Poisson process with rate . The levels of the QBD corresponds to the queue length (number of packets waiting to be served). There are dim **D0** 1 M states for the MAP and only 1 state for the Poisson process, so the total number of phases for the QBD is M 1. The arrival and the service distributions does not depend on queue length, so the chain behaves the same at every level, except when empty 0 it cannot serve packets: **B** **B** **F** **F** 0 and **L** **L** 0. For this MAP/M/1 QBD the transition block matrices are:

- **B** **I**, the backward transitions i 1 j are determined purely by the service demand distribution
- **F** **D1**, the forward transitions i 1 j are fully characterized by the MAP arrival process' type 1 transitions and
- **L** **D0** **I**, **L** ^{0} **D0**, the local phase transitions i j i j are determined by the MAP arrival process' type 0 transitions.

Note that **I** is the identity matrix with dimension M M and for **Q** to be a valid generator, **I** has to be deduced from **L**.

The evolution of a continuous time Markov chain X t with state space S K S , generator **Q** are described by the transition probability functions **P** t $e^{\mathbf{Q}t}$. Note that **P** t is a matrix

DOI:10.15774/PPKE.ITK.2019.006

function. One element $i\ j$ of $\mathbf{P}\ t : P_{ij}\ t \quad X\ t \quad j X\ 0 \quad i$ describes the probability that for a time interval t the chain was in state i at the beginning of the interval, but it is in state j at the end of the interval. Consequently the i^{th} row describes the evolution of state transition probability from state i to any other state.

$$\mathbf{P}\ t \quad \begin{array}{|cccccccccc|} \hline X\ t & 1 X\ 0 & 1 & X\ t & 2 X\ 0 & 1 & & X\ t & K X\ 0 & 1 \\ X\ t & 1 X\ 0 & 2 & X\ t & 2 X\ 0 & 2 & & X\ t & K X\ 0 & 2 \\ \multicolumn{3}{|c}{\vdots} & \multicolumn{3}{c}{\vdots} & \ddots & \multicolumn{3}{c|}{\vdots} \\ X\ t & 1 X\ 0 & K & X\ t & 2 X\ 0 & K & & X\ t & K X\ 0 & K \\ \hline \end{array} \tag{2.86}$$

The Kolmogorov forward equations, which describe the time dynamics of the probability transitions take the form:

$$\mathbf{P}\ t \quad \mathbf{P}\ t \quad \mathbf{Q} \quad \mathbf{P}\ 0 \quad \mathbf{I} \tag{2.87}$$

where $\mathbf{P}\ t$ is the time derivative of each transition probability function. One could fully characterize the chain if these set of differential equations are solved. For example the necessary probabilities for the information theoretic metrics could be computed as:

$$\hat{p}_i \qquad \text{queue len. is in interval } i \text{ at time } t_0 \quad t \text{ queue len. was out of interval } i \text{ at } t_0 \tag{2.88}$$

$$X\ t_0 \quad t \quad k X\ t_0 \quad m \quad X\ t_0 \quad m \tag{2.89}$$

$$m\ I^i\ k\ I^i$$

$$\tilde{p}_i\ j \qquad \text{queue len. is } j \text{ in interval } i \text{ at time } t_0 \text{ queue len. is in that interval} \tag{2.90}$$

$$\frac{X\ t_0 \quad j}{k\ I^i \quad X\ t_0 \quad k}\ j \quad I^i \tag{2.91}$$

The dynamical behavior of a chain is usually split into two parts: the transient part and the steady state part. The steady state (long run average) behavior of a chain is characterized by the stationer distribution (row vector). Throughout this discussion we assume that this distribution exists, as it does in most practical applications. The stationer distribution satisfies the following conditions:

$$\mathbf{P}\ t \quad t \quad 0 \tag{2.92}$$

$$\mathbf{0} \quad \mathbf{Q} \tag{2.93}$$

$$i \quad 1 \tag{2.94}$$

$$i$$

where (2.93) are called the global balance equations. Equation (2.92) describes that if the queue is in steady state, the transition probabilities do not evolve further. This means that over a long period of time we find the chain in state i with probability $_i$. Also it can be shown that the limiting probabilities converge to for every row of $\mathbf{P}\ t$:

$$\lim_t \mathbf{P}\ t \quad \begin{array}{|c|} \hline \\ \hline \\ \hline \vdots \\ \hline \\ \hline \end{array} \tag{2.95}$$

which means on the other hand (by the definition of $\mathbf{P}\ t$), that from any state i (i^{th} row) the chain

DOI:10.15774/PPKE.ITK.2019.006

transitions (on the long run) to state j with probability $_j$. If we know the steady state distribution of chain $X\ t$, and assume that the chain converges fast to this distribution, we can compute the following probabilities:

$$\begin{aligned} \hat{p}_i & \quad X\ t_0 \quad t \quad k\, X\ t_0 \quad m \quad X\ t_0 \quad m \\ \hat{p}_i & \quad k \quad m \end{aligned} \tag{2.96}$$

where the first term became $_k$ because of (2.95) and the second term became $_m$ because of (2.92).

$$\begin{aligned} \tilde{p}_i\ j \quad & \frac{X\ t_0 \quad j}{k\ I^i \quad X\ t_0 \quad k} \quad j \quad I^i \\ & \frac{j}{k\ I^i \quad k} \quad j \quad I^i \end{aligned} \tag{2.97}$$

Since the MAP/M/1 queue is a QBD process which is governed by an underlying continuous time Markov chain, if we can compute its stationer distribution, we can derive the required probabilities to compute the both the Signaling Entropy and the Link Entropy metrics. The stationer distribution of a QBD can be computed effectively using the matrix analytic methods described in [122, 124, 123, 135, 9, 141]. A Matlab implementation of these algorithms (and much more) can be found in the Q-MAM, SMCSolver, MAMSolver packages [133, 134, 13, 149, 148, 110, 137, 74, 46, 47, 95].

The Signaling Entropy and the Link Entropy can be written as follows:

$$\mathrm{H} \quad _{u\,v} \quad \mathbf{D0}\ \mathbf{D1} \quad t \quad _{u\,v} \quad t \tag{2.98}$$

$$\mathrm{H} \quad _{u\,v} \quad \mathbf{D0}\ \mathbf{D1} \quad _{u\,v} \quad \mathbf{D0}\ \mathbf{D1} \quad t \tag{2.99}$$

where parameters $_0 \quad _1 \quad _{L\ 1}$, and $\mathbf{D0}\ \mathbf{D1}$ belong to the MAP/M/1 queue describing the traffic on link $u\ v$. Now optimal link scaling can be posed as a constrained optimization problem:

$$\begin{aligned} & \min_t \mathrm{H} \quad _{u\,v} \quad \mathbf{D0}\ \mathbf{D1} \quad _{u\,v} \quad \mathbf{D0}\ \mathbf{D1} \quad t \\ & \text{s.t.}\ \mathrm{H} \quad _{u\,v} \quad \mathbf{D0}\ \mathbf{D1} \quad t \quad _{u\,v} \quad t \end{aligned} \tag{2.100}$$

Based on this calculation a simple search can yield (under checking the constraint in each step) the optimal solution which maximizes the routing performance under the constraint of keeping the signaling bandwidth below a threshold.

THESIS I.4 (optimizing link scaling using MAP/M/1)**.** *In* (2.100)*, I formulated a constrained optimization problem which connects the information about the random link descriptors (Link Entropy) and the appropriate bandwidth of the signaling process to support that information*

DOI:10.15774/PPKE.ITK.2019.006

(Signaling Entropy) at a certain probability. I proposed a computable solution to this problem by modeling the dynamics of the link descriptors as MAP/M/1 described in (2.98) *and* (2.99)*, Consequently the information theoretical quantities can be obtained analytically and the optimal solution can be found.*

The thesis is restated in a self consistent way in Appendix A at Thesis I.4 (page 86).

2.6 Simulations and numerical results

The simulation results are presented and ranked for the performance of the previously defined algorithms. The effect of "signaling entropy" on "link entropy" is studied based on real life traffic.

2.6.1 Routing in unicast case

In this section the performances of the newly developed QoS routing algorithms and link optimization methods are analyzed by extensive simulations. In order to compare the algorithms I introduce a performance measure for comparing two paths R_a and R_e both starting from node *src* and ending at node *dst*. This is defined as the ratio of the probability of the path R_e (found by the exhaustive search) and the probability of the path R_a (found by a given algorithm) fulfilling the end-to-end QoS criterion, given as follows:

$$R_a \; T \; : \qquad \underset{u\,v \quad R_e}{\qquad} \; {}_{u\,v} \quad T \qquad \underset{u\,v \quad R_a}{\qquad} \; {}_{u\,v} \quad T \tag{2.101}$$

Given that R_e is the best route in the sense that it fulfills any T QoS criterion with the largest probability, $R_a \; T \quad 0$, and it measures the "performance drop" in probability for given T QoS value. An example is given at Figure 7, where on the left side the three curves correspond to three different paths: $R_{a1} \; R_{a2} \; R_e$ respectively. On the right side the two bell shaped curves correspond to $R_{a1} \; T$ and $R_{a2} \; T$. The normalized area under these curves are defined as:

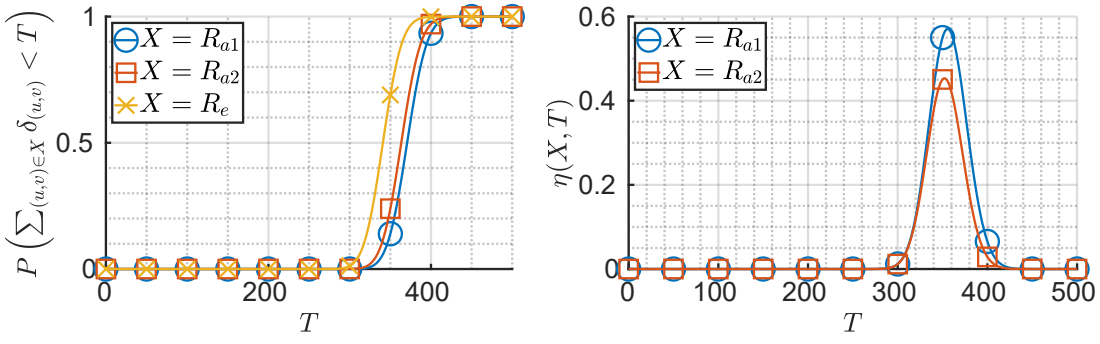


Figure 7: An example for

DOI:10.15774/PPKE.ITK.2019.006

$$R_a \quad \frac{\int_0^{T_{\max}} R_a \; T \; dT}{\int_0^{T_{\max}} \quad u\,v \quad T \quad dT} \qquad (2.102)$$
$$0 \qquad u\,v \quad R_e$$

where $T_{\max}$ is the investigated maximum value for T. corresponds to the performance drop of a path compared to the best path. It is easy to see that the "worst path" R_0 which can fulfill any T $T_{\max}$ only with 0 probability corresponds to the value R_0 1 and the best path R_e corresponds to R_e 0. The closer this function approximates the value 0, the better the performance of the corresponding route is. If we investigate these metrics on an ensemble of measurements we can characterize the average performance of the proposed routing algorithms.

Figure 8 depicts the test network, which was used to simulate the performance of the different routing algorithms. The network model is based on the European part of the GEANT[48] network topology. The aggregator switches in Russia were removed to lower the edge and node counts, so the exhaustive search could be executed in a feasible time. This particular topology has 46 nodes and 136 edges. I have generated the delay on each link based on its own Poisson

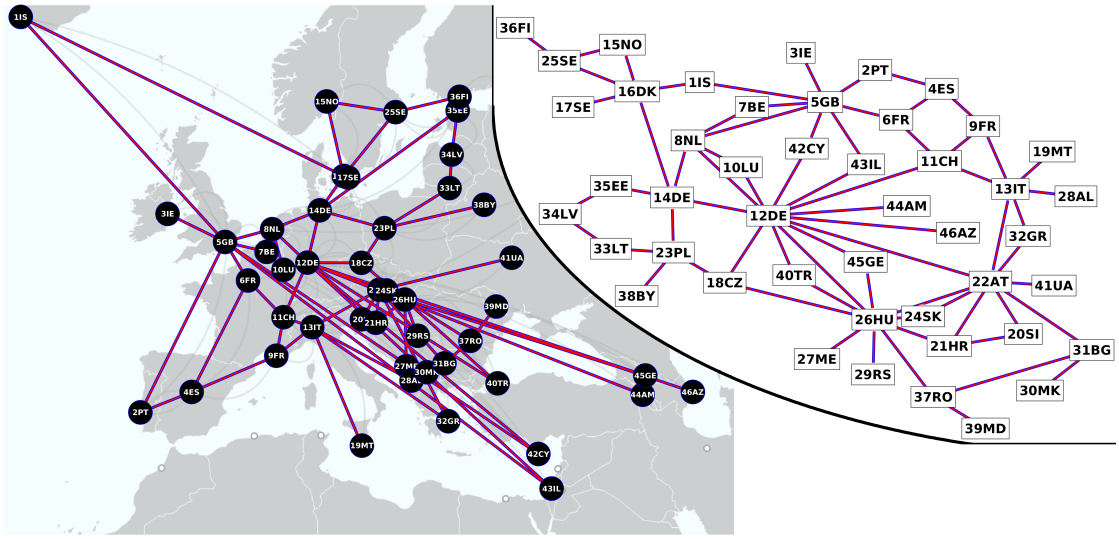


Figure 8: GEANT network topology used to evaluate the proposed algorithms. On the left side the Mercator projection on the right side a flattened representation of the topology can be seen.

distribution. I chose each link's parameter randomly from the range 0 to 230 uniformly. Each link had a LAS with 10 divisions according the the "Gaussian approximation" algorithm: LAS 0 5 16 32 53 79 110 146 187 233 284 . Figure 9 depicts an example delay distribution ensemble.

The performance of four algorithms will be shown and compared to the performance of the exhaustive search: "OSPF","Gaussian approximation","exhaustive-s","recursive-s" The "OSPF" is a simple shortest path algorithm having a metric as the advertised LAS entry. The "Gaussian approximation" had its metric according to Theorem 1, the "exhaustive-s" is based on Algorithm 1 and "recursive-s" is based on Algorithm 2.

DOI:10.15774/PPKE.ITK.2019.006

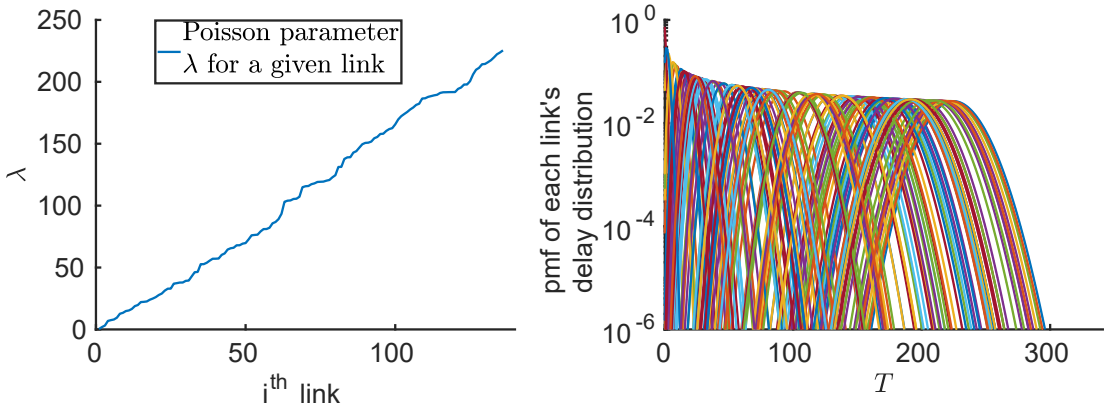


Figure 9: Example delay distributions on each link

I have chosen Island "1ISL" for the *src* node and Romania "37RO" for the *dst* node, as this pair offers a wide variety of routes in between them. All the routes from *src* to *dst* can be labeled with an index (1 to 4937) in an increasing order according to R_a . This labeling can be seen in Figure 10 Using the fixed and known link descriptor distribution (Figure 9) I sampled the

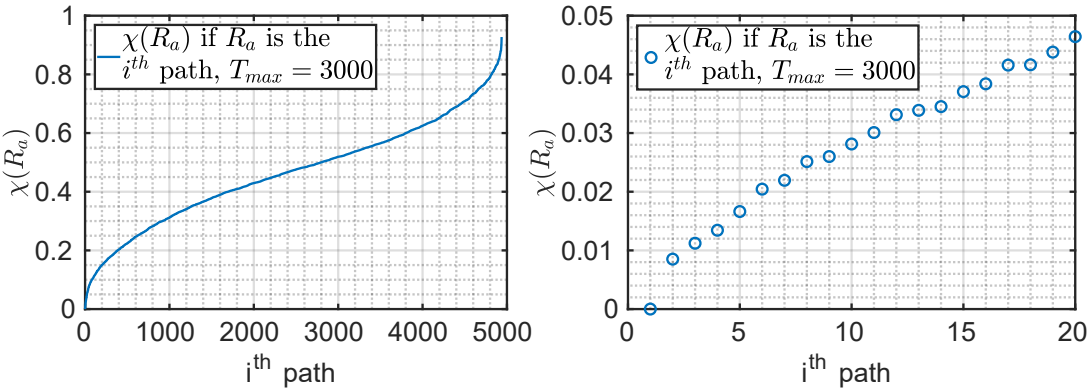


Figure 10: Path index vs its performance , T_{max}=3000. The left figure depicts all the paths, while the figure on the right zooms in to the first 20 best path.

network. When the OSPF algorithm found the best route (path #1) so did all other algorithms. For this particular link distribution ensemble the relative frequency for the OSPF not choosing path #1 was P 0 4148, so almost half of the time there were better performing paths. To quantify the improvement of the introduced algorithms I chose 10000 sample points when the OSPF algorithm did not choose path #1. The frequency of the routes chosen by the algorithms are depicted in Figure 11. From this figure and from the previous statement it can be seen that all introduced algorithms perform better than the OSPF. The exhaustive-s algorithm performs the best as it has access to the most precise information about the network, although it has the highest computational complexity. The Gaussian and the recursive-s algorithm performs nearly identically and both have similar computational complexity. This gives the upper hand to the recursive-s algorithm, because the Gaussian algorithm has a rather strict constraint on the chosen LAS shape while the recursive-s can be utilized for any type of LAS.

DOI:10.15774/PPKE.ITK.2019.006

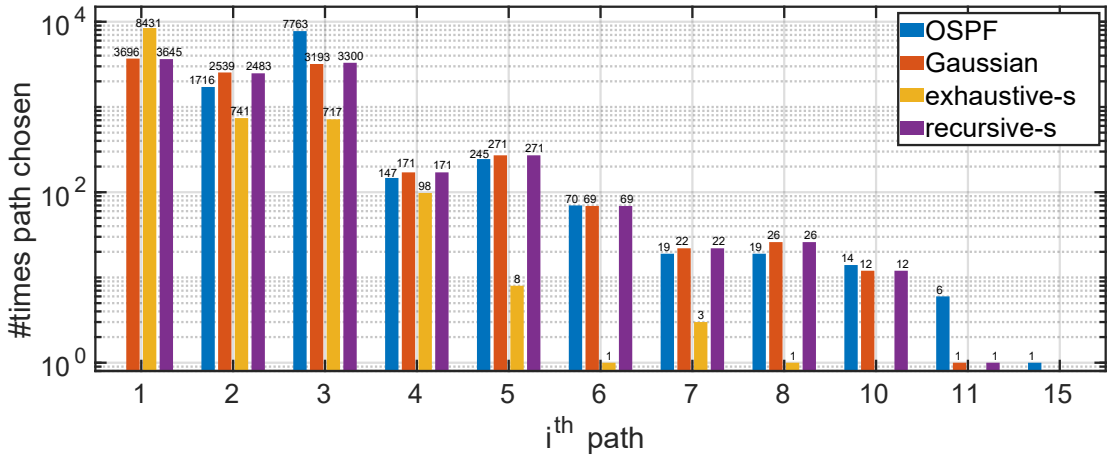


Figure 11: Path choice frequency out of 10^4 samples when the OSPF did not choose path #1

In order to obtain more general numerical results, the algorithms were run not only on one graph with particular link states, but on a set of graphs denoted by . Each graph had 10 nodes having cardinalities at least 3, and the random graph generator made sure that all nodes has belonged to the same component. Let us denote all possible routes (from all possible *src* to all possible *dst*, *src* *dst*) in a graph G with G. I characterize the "ensemble" performance by metrics G and $_e$, given as follows:

$$G : \frac{1}{G} \quad R \quad _e : \frac{1}{G} \quad G \tag{2.103}$$

The distribution of the number of paths from all *src* to all *dst* in can be seen on Figure 12. From

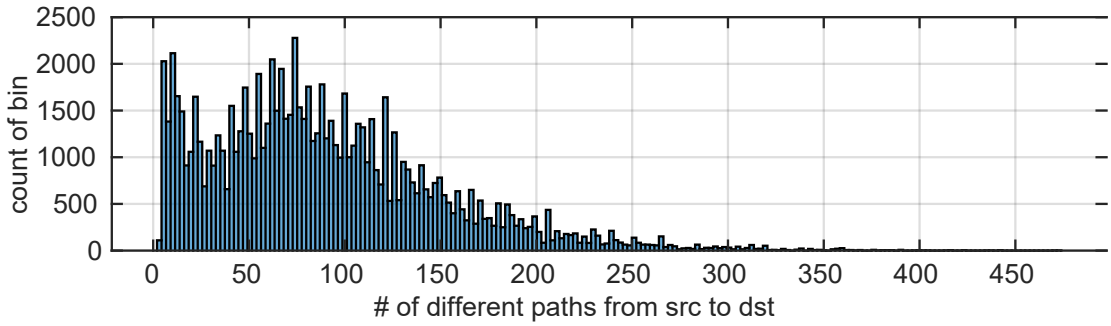


Figure 12: The distribution of the number of different paths between all *src* to all *dst* for the random graph ensemble

this figure it can be seen that there were almost always multiple choices for the algorithms to choose a path from. For this ensemble the distribution of G is depicted in Figure 13 and the total ensemble performance metric $_e$ is in Figure 14. From these figures it can be seen that on average all novel algorithms outperform the OSPF. The recursive-s and the Gaussian algorithms perform nearly identically. Both have similar computational complexity, but the Gaussian algorithm is limited by the strict rule for the choice of the LAS, while the recursive-s is not. The exhaustive-s algorithm performs the best, although it has the highest computational complexity and requires the link descriptor to be to be modeled precisely.

DOI:10.15774/PPKE.ITK.2019.006

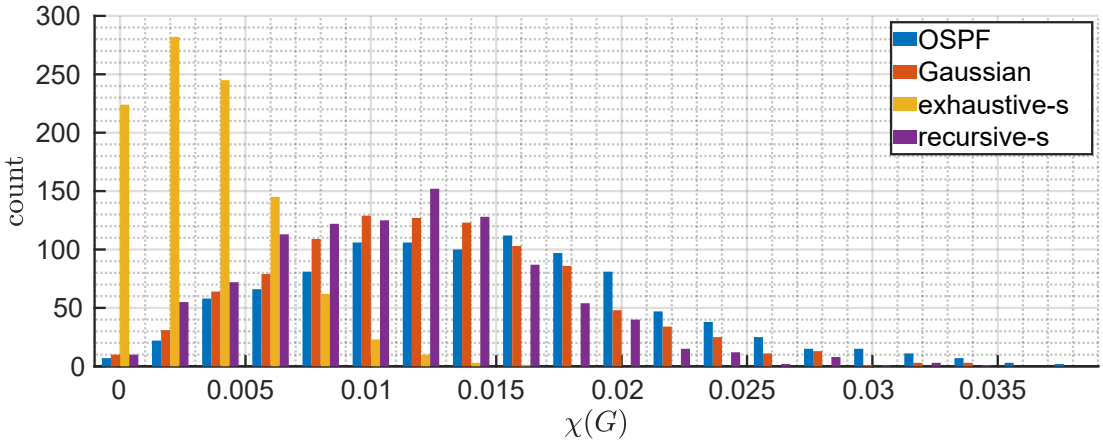


Figure 13: The distributions of the ensemble performance metric G over all graphs G in the random graph ensemble for all for all introduced algorithms.

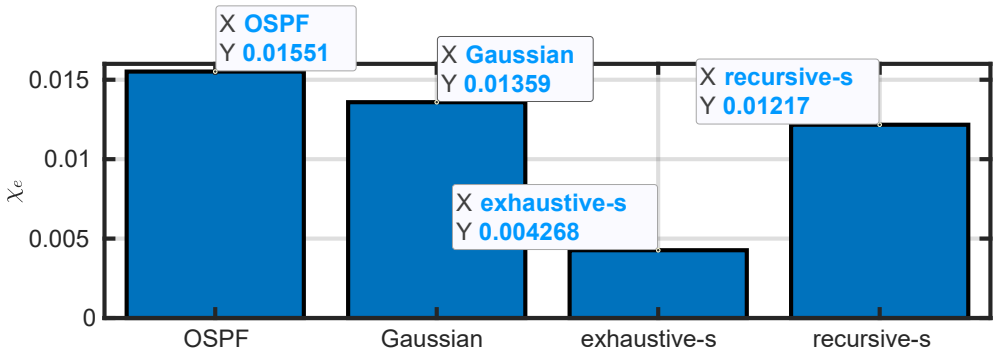


Figure 14: The total ensemble performance metric $_e$ for all introduced algorithms.

Based on the simulation results the developed algorithms can be ranked with respect to their performances as follows: 1: “Recursive-s Algorithm”, 2: “Exhaustive-s Algorithm”, 3:“Gaussian approximation”, 4:“OSPF”. For practical applications either the “Recursive-s” (low computational complexity) or the “Exhaustive-s” algorithm (good performance, higher computational complexity, need for good link descriptor model) is is recommended.

DOI:10.15774/PPKE.ITK.2019.006

2.6.2 Multicast routing with incomplete information

The $\tilde{A}_1$ and the $\tilde{A}_2$ objective functions (defined in (2.4) (2.5)) were evaluated by exhaustive search and the HNN based algorithm on a graph with the following parameters: The size of the network N 8, the Rayleigh channel parameters were chosen to typical or better indoor environment: 3 g 1 10 s^2 1. The positions of the nodes were randomly generated according to i.i.d. uniform distributions in the unit square. The group of the multicast nodes consisted 3 randomly chosen nodes.

I have performed the exhaustive search by enumerating all the possible trees and evaluating the objective functions on the trees. I have compared the results of the HNN algorithm to the exhaustive solution. For the $\tilde{A}_2$ objective function I have evaluated the performance given by the Chernoff bound and also the corresponding theoretical probability by performing convolutions on the known distributions.

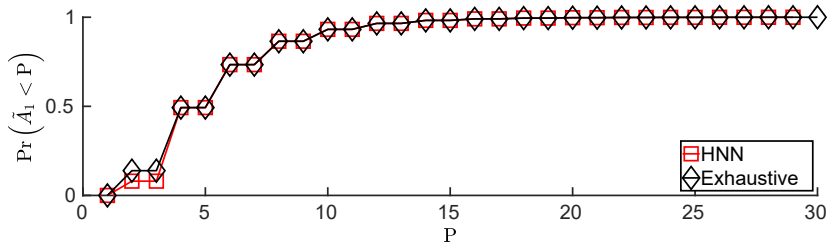
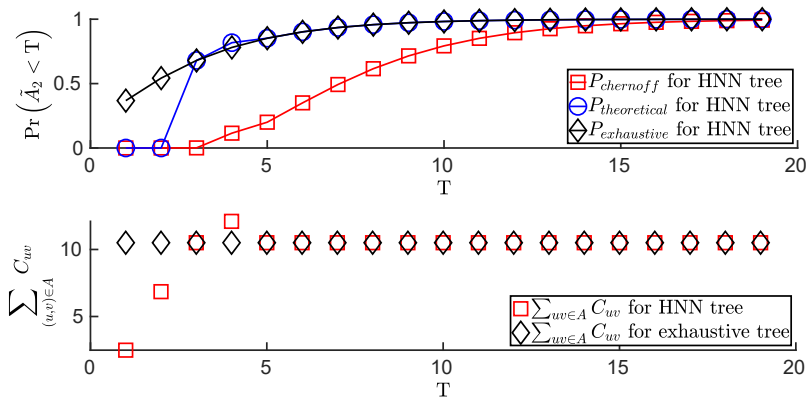


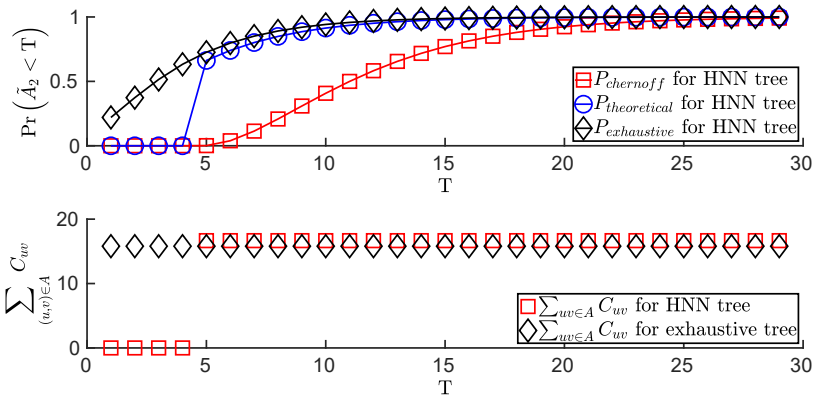
Figure 15: A typical evaluation of the $\tilde{A}_1$ objective function.

It can be seen in Figure 15 that the HNN algorithm can find almost always the optimal solution for the $\tilde{A}_1$ objective function of the bottleneck problem: Pr $\tilde{A}_1$ P : max $_{u\,v}$ A $_{u\,v}$ P . This figure is typical in the sense that throughout the simulation runs I have seen the same behavior. For the $\tilde{A}_2$ objective function the figures show the probabilities of meeting the delay constraint and the energy consumption for the tree of choice: Pr $\tilde{A}_2$ T : max$_{R_{src\,m}}$ A $_{u\,v}$ $R_{src\,m}$ $_{uv}$ T $_{u\,v}$ A C_{uv} In Figure 16a a case can be seen at T 4 that the HNN finds a solution that satisfies the delay constraint with a higher probability in the expense of larger transmit power. For larger values the solution given by the HNN is the same as the optimal solution given by the exhaustive search. In Figure 16b for small T values it can happen that individual link measures approximated by the Chernoff bound could not give a positive probability of meeting the delay constraint, hence the HNN could not supply a valid tree. However solutions exist in that region which is not found due to the un-sharpness of the Chernoff bound.

DOI:10.15774/PPKE.ITK.2019.006



(a) Near optimal solution for $\tilde{A}_2$ obj. func.



(b) A typical evaluation of the $\tilde{A}_2$ obj. func.

Figure 16: Performance of the multicast tree finder algorithm in case of additive measures

2.6.3 Link scaling

I have based my traffic models on the publicly available DISCO data-set from the Measurement-lab data-set [115]. Specifically I choose the switch connected to the "mlab1.dub01.measurement-lab.org" server. Since June 2016, M-Lab has collected high resolution switch telemetry for each M-Lab server and site uplink.[114] I have used the "switch.octets.local.rx" and "switch.unicast.local.rx" metrics from the data-set, which correspond to the "Bytes received by the machine switch port" and "Unicast packets received by the machine switch port" to gather the necessary statistics for the traffic models. The data-set contains these metrics with sampling time of 10s. Real traffic has self similarity in several levels, one of them is a daily self-similarity.

I assumed that within an hour interval the traffic somewhat stays the same (relative to the daily regular fluctuations). Based on this I have chosen two time periods over which I have aggregated the necessary statistics to derive the traffic models.

- An average traffic load: from 14:00-15:00 each day in 2018. May. 1. to Jun. 30.
  Traffic situation 1 has a mean packet rate of 6066 341pps, std:6865 61pps, max:75944 03pps and mean speed 62Mbps, std:78 28Mbps, max:883 66Mbps.
- A more intensive traffic load: from 18:30-19:30 each day in 2018. Sep. 1. to Nov 30.
  Traffic situation 2 has a mean packet rate of 13613 81pps, std:12278 3pps,

DOI:10.15774/PPKE.ITK.2019.006

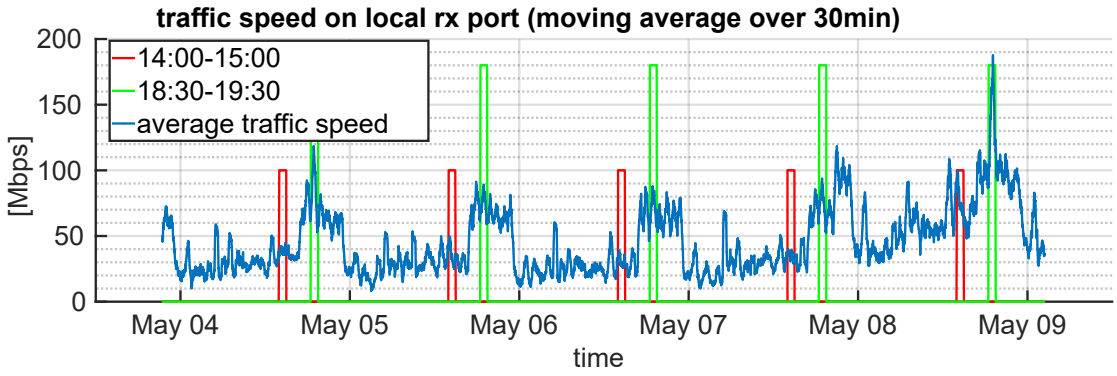


Figure 17: Example of the daily self similarity pattern in the real switch telemetry data from mlab1.dub01.measurement-lab.org. Moving averaged data was plotted from 2018. May 4-9.

max:137180 7pps and mean speed 143 5Mbps, std:141 5Mbps, max:1605 45Mbps.

For the traffic models I derived from the data I assumed that the inter-arrival times in a 10s slot follows an exponential distribution with rate corresponding to the metric in that 10s slot. Based on the gathered packet count statistics from metric "switch.unicast.local.rx" I derived the average packet rates (packet/second) for each 10s slot. From this I derived the corresponding average inter arrival times. This was fed into an event generator which generated events according to the specified average inter arrival times. This sequence of events were fed into the MAP estimator of the kpc-toolbox by [46, 47, 95]. I have used 16 state MAP-s in both cases to model the sequence of events. From the identified models I also generated sequences of events, which then were mapped back to the average packet rate metric for comparison. The statistics of the real world traffic and the identified traffic models for both scenarios are depicted in Figure 18 and Figure 20 with the average packet sizes in Figure 19 and Figure 21 respectively. Please note that since the traffic statistics were quite similar to the "exponential" distributions, the figures have logarithmic X axes and the histogram bins were generated logarithmically to emphasize the mean characteristics. From these figures it can be seen that the identified models are detailed enough to reproduce the statistics of the real life data.

Since there is no information available on the used switches, I assumed that the local rx link was an 10GBASE-X connection, since the data-set contains data points with speeds greater than 1Gbps. From the modeling point of view this means that the traffic is bottlenecked by the connected router's packet processing performance. It can be seen from Figure 19 and Figure 21 that the majority of the traffic flows through the router with packet sizes close to the MTU, when computing the average packet processing speed of the router I assumed that the average packet was 1450 bytes long. I also assumed that the router's average packet processing data rate is 1Gbps for the sake of presenting the numerical results, but it could have been adjusted as desired.

Having the MAP models of the incoming traffic and the average packet serving rates of the router all parameters (**D0 D1**) are available for the MAP/M/1 model to be analyzed according to subsection 2.5. Based on these values modeled system 1 had load 0 0703 and modeled system 2 had load 0 1579 The required metrics were calculated using the toolboxes Q-MAM, SMCSolver, MAMSolver [133, 134, 13, 149, 148, 110, 137, 74, 46, 47, 95].

DOI:10.15774/PPKE.ITK.2019.006

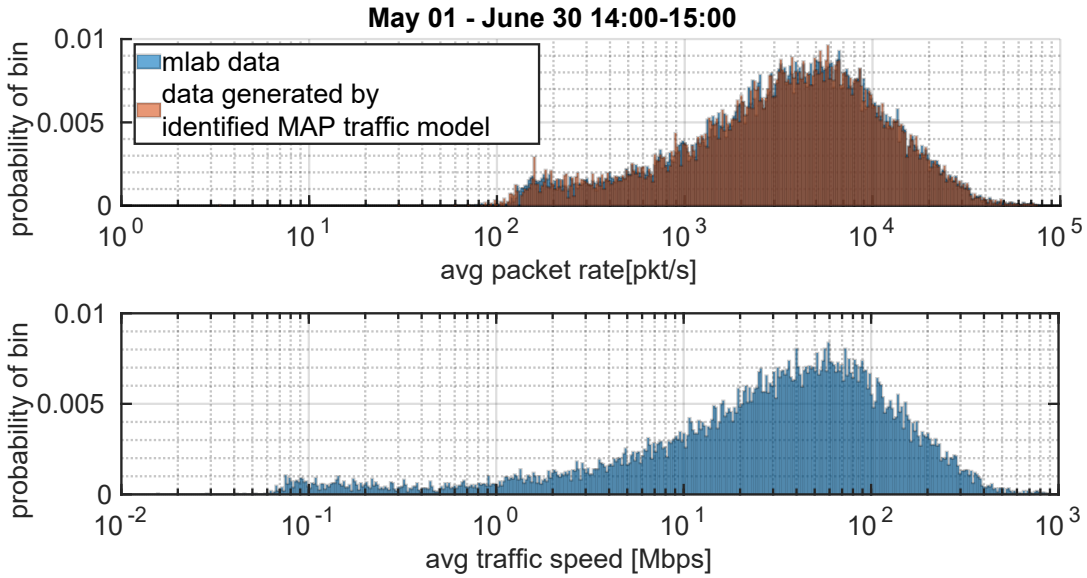


Figure 18: Packet rate and speed statistics aggregated over the time interval 14:00-15:00 each day in 2018. May. 1. to Jun. 30 and generated from the corresponding model

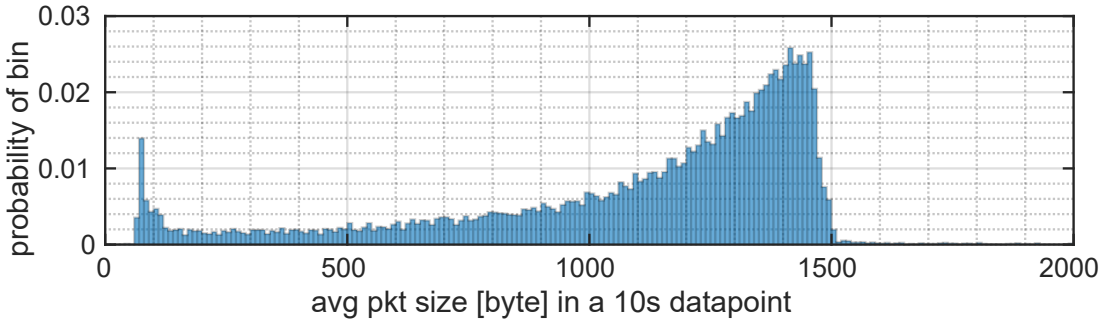


Figure 19: Packet length distribution aggregated over the time interval 14:00-15:00 each day in 2018. May. 1. to Jun. 30

For computing the information theoretic metrics, two types of link schemes were used: an equidistant type and an exponential type.

$$\mathrm{LAS}_{\mathrm{equidistant}}: \quad t_i \quad ceil\ i \quad t \quad i \quad 0 \quad \frac{L}{t} \tag{2.104}$$

$$\mathrm{LAS}_{\mathrm{exponential}}: \quad t_i \quad \begin{matrix} 0 & & & i & 0 \\ round & \exp & i & t & i & 1 \end{matrix} \quad \frac{\log L}{t} \quad t \quad \log 2 \tag{2.105}$$

On Figure 22 and Figure 23 one can see the SE (2.98) and the LE (2.99) plotted against the link scaling. Here the link scaling means the number of divisions over the link descriptor.

In both cases can be seen that the exponential grid for this type of traffic is a better choice, since when achieving a similar LE values (red curve and purple curve Y values) the corresponding SE is lower for the exponential grid (green curve vs blue curve Y value). Also it can be seen that a higher load on the system causes the number of packets to be served to fluctuate more, which results

DOI:10.15774/PPKE.ITK.2019.006

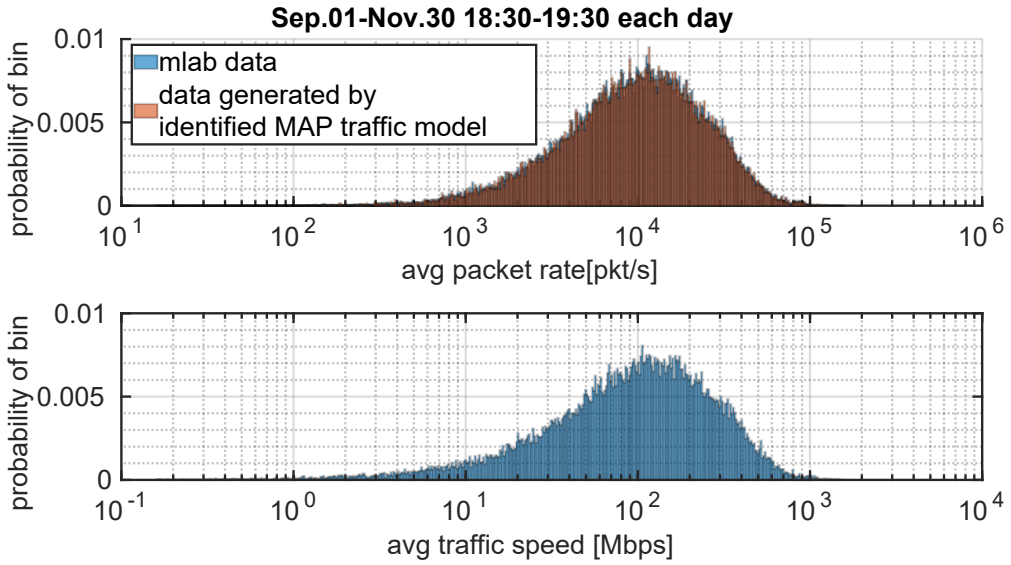


Figure 20: Packet rate and speed statistics aggregated over the time interval 18:30-19:30 each day in 2018. Sep. 1. to Nov 30 and generated from the corresponding model

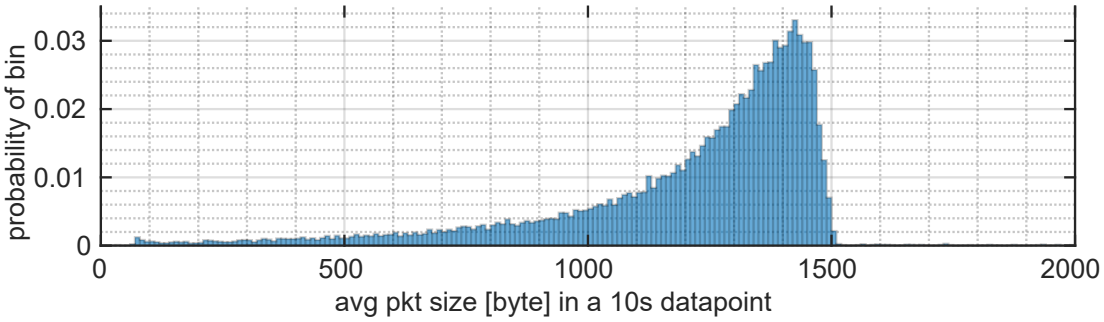


Figure 21: Packet length distribution aggregated over the time interval 18:30-19:30 each day in 2018. Sep. 1. to Nov 30

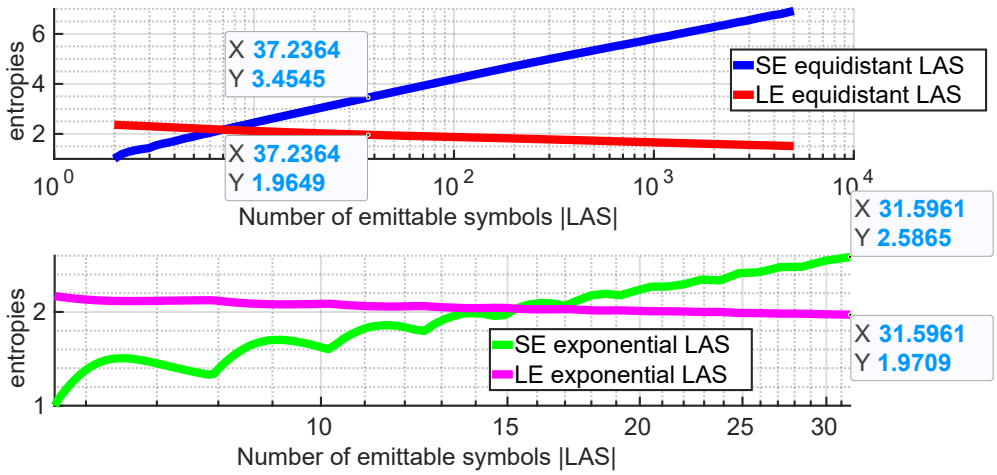


Figure 22: Information theoretic metrics for traffic situation 1

DOI:10.15774/PPKE.ITK.2019.006

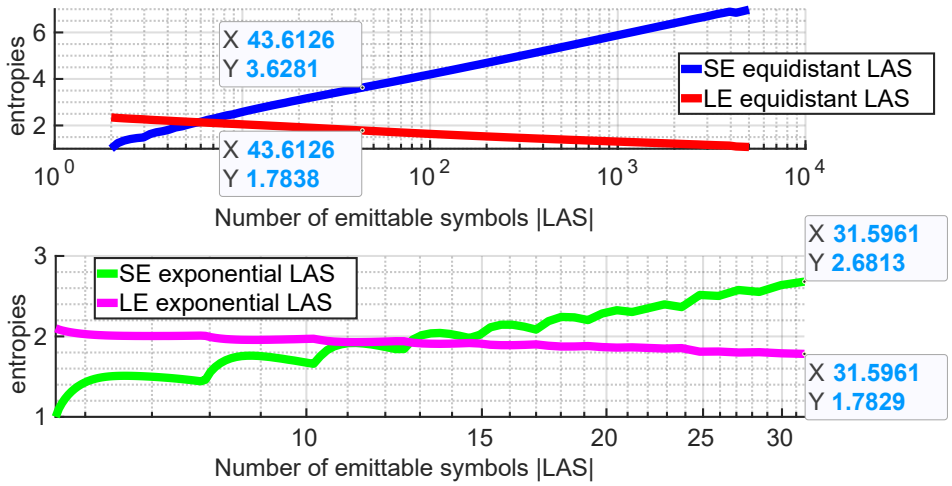


Figure 23: Information theoretic metrics for traffic situation 2

in an elevated number of advertisements which in turn shows up in an increased SE. One can see that increased link scaling(less division) indeed will raise the number of misidentified paths selected by the routing algorithm (due to the less complete link state information) and at the same time will reduce the signaling bandwidth necessary for link state advertisement. Based on these figures engineering design can be employed by setting the threshold on the signaling bandwidth and then reading out the obtained performance. As a result, one can analyze the trade-off between routing performance and signaling bandwidth.

2.7 Conclusion

In this section new algorithms were proposed to carry out unicast QoS routing with incomplete information. The proposed algorithms are capable of carrying out routing in polynomial time. Based on the theoretical and numerical analysis the best method is the General Normal algorithm, however, methods based on the Chernoff inequality also provide good performance. For multicast scenarios even for networks sizes as small as 20 nodes exhaustive search is unfeasible so heuristics are needed to approximate a good solution. I have shown that a HNN based heuristics with a properly chosen additive measures can yield to a good solution for this traditionally NP complex problem. Because of the conservativeness of the Chernoff approximation, the delay bound is always met in the expense of consuming more transmit power. Because of the large free parameter set and the contradictory constraints for constructing the energy function the HNN may not be the best method for solving this quadratic optimization problem. Other heuristics like applying SDR could be considered to be an alternative. Furthermore, in order to optimize link scaling information, theoretical measures were introduced which can maximize routing performance under the constraint of keeping signaling information bellow a threshold. In this way, optimal bandwidth utilization can be achieved in packet switched networks. As the simulation results have indicated the choice of link advertisement interval (t) has a great impact on QoS.

DOI:10.15774/PPKE.ITK.2019.006

3 Thesis group II - a heuristic solver based on hypergraphs for UBQP and its applicability in ICT

This chapter is organized around the combinatorial approach and tractability of several ICT problems, since they can be formulated as an Unconstrained Binary Quadratic Programming (UBQP). These applications include load balancing, a wide class of scheduling problems, MUD, VLSI design, Steiner tree problems...etc. A survey of these applications can be found in [92].

In cloud computing environments and in IoT the efficient scheduling [109, 166] and distribution of tasks plays a central role in performance and scalability. Several approaches exist [139, 165, 147] - usually metaheuristics - to address these problems but at their core almost each of them contains a method for approximating a solution of a constrained optimization problem. This core step can be usually formulated as a UBQP, therefore the proposed algorithms can be utilized on it. Recent surveys on scheduling in IaaS cloud computing environments and load balancing can be found in [139, 165, 147]

Furthermore, other problems under linear constraints can also be transformed into UBQP as demonstrated in [91, 11, 118, 163, 27, 108]. Unfortunately, UBQP has proved to be NP-hard [45], but in some special cases it can be solved in polynomial time [11, 127, 128, 7]. In general though, there is still a great need for developing fast methods which can reach near-optimal solutions when the size of the problem goes beyond a given limit. Thus the aim of this chapter is to present some novel approaches to UBQP which are based on recursive dimension reduction (or addition) techniques. Although the more complex applications are more relevant, the proposed algorithms and their performance will be presented in detail on simpler applications for traceability:

- large scale problems listed in ORLIB.
- simple scheduling;
- Multiuser Detection;

Based on the performance analysis the new algorithms prove to be superior to the known heuristics regarding both the quality of the achieved solution and the convergence time. *The corresponding publication of the author is titled "Novel algorithms for quadratic programming by using hypergraph representations" [160].*

This chapter is organized as:

- in subsection 3.1, the related work is summarized;
- in subsection 3.2, the formal model is outlined;
- in subsection 3.3, the new algorithms are detailed;
- in subsection 3.4, the new methods are tested on large scale problems selected from ORLIB;
- in subsection 3.5, the application to scheduling is elaborated followed by some numerical results;
- in subsection 3.6, the application to MUD is detailed followed by a performance analysis;
- finally, in subsection 3.7, some conclusions are drawn.

DOI:10.15774/PPKE.ITK.2019.006

3.1 Related work

UBQP has been treated by many researchers in the past decades. There are exact methods developed for solving it, but beyond a given size the complexities of these methods tend to become prohibitive because Garey and Johnson (1979) proved that UBQP in general is an NP hard problem [45]. Therefore, there are well-known heuristics which are applied to large scale problems. These heuristics usually apply different strategies [54] or combine several methods, such as LS [117, 15], TS [126, 91, 11, 53], SDR [129], SA [11], GA, EA [106], MA [118] HNN [151, 163].

First we give a brief historical overview of the methods applied to UBQP. Hopfield et al. (1985) applied the Hopfield network on the TSP problem to obtain a sub-optimal solution. At that time the term UBQP was not coined. Barahona et al. (1986) proved that some special classes of UBQP can be solved in polynomial time [7]. Boros et al. (1989) introduced the DDT exact method to solve UBQP which transformed UBQP into a polynomial PBF. The DDT based solution still enjoys a great deal of popularity [61]. Poljak et al. (1995) proposed relaxation techniques by using SDR, linearization in order to limit the problem complexity [129] and they present the applicability on QAP, graph partitioning problems and to the max-clique problem. Helmbert et al. (1998) used SDR in combination with CP and analyzed BB algorithms on 100 400 magnitude problems [68]. Glover et al (1998) developed methods to use the TS with adaptive memories and applied to problems of magnitude 100 500 [53]. Beasley et al. (1998) investigated TS and SA [11] applied to problems in ORLIB. This was the fist study to incorporate a public and comparable test set to the UBQP problem. Smith et al. (1998) used neural network solutions (e.g. HNN and SONN) for solving CSP which was modeled as a UBQP [150]. Simth et al. (1999) gave a survey on the application of neural networks on COP including UBQP [151]. Merz et al. (1999) investigated GA with hybrid LS and applied onto problems of magnitude 200 2500 [116]. Lodi et al. (1999) used EA heuristics [106] for problems up to 500 variable and compared them to algorithms like TS and BB. Glover et al. (2002) used a “one pass” heuristics based on the DDT algorithm on problems up to 9000 variables [54]. Merz et al. (2002) used greedy 1-opt and k-opt LS heuristics on problems of 10 2500 magnitude [117]. Kochenberger et al. (2004) introduced a number of transforms of COP to a unified UBQP and he tested TS and SS algorithms on problems like K-coloring and Max Sat [91]. Merz et al. (2004) use MA which is a hybrid algorithm combining EA and LS. This algorithm was tested up to 2500 variables [118]. Xia et al. (2005) use DHNN on the problem of operating a crossbar switch in an efficient way on problem sizes of 20 2000 which is a practical application of the UBQP formulation [164]. Azim et al. (2006) apply HNN to QAP and GPP problems [5]. Palubeckis et al. (2006) investigates the usage of ITS heuristics up to 7000 variables [126] Alain et al. (2007) shows relaxation techniques for MIQP solvers and apply it to UBQP for different density problems of 50 200 magnitude [12]. Luo et al. (2010) published a summary paper for solving QP by using randomized SDR [108], while Wang et al.(2010) improves the solution by using HNN and EDA [163]. Chicano and Alba (2011) investigated the difficulty of a UBQP with an elementary landscape decomposition technique [21]. But the methods proposed in these papers still did not strike a good compromise between complexity and the quality of achieved solutions in large scale problems.

DOI:10.15774/PPKE.ITK.2019.006

3.2 UBQP formulation

In this section we introduce the mathematical framework of the problem together with a graph based representation of the problem. Furthermore we summarize the key steps of some well-known algorithms from the graph based perspective.

The UBQP is a quadratic COP where each component of vector **y** can have two distinct values, which are taken to be -1 and +1 in the forthcoming analysis.

$$\mathbf{y}\ \mathbf{W}\ \mathbf{b}\ :\quad \mathbf{y}^T\mathbf{W}\mathbf{y}\quad 2\mathbf{y}^T\mathbf{b} \tag{3.1a}$$

$$\mathbf{y}\qquad 1\ ^N\ \mathbf{b}\qquad ^N\ \mathbf{W}\qquad ^N\ ^N \tag{3.1b}$$

$$\mathbf{y}_{opt}\qquad \min_{\mathbf{y}\quad 1\ ^N}\qquad \mathbf{y}\ \mathbf{W}\ \mathbf{b} \tag{3.1c}$$

The following assumptions can be used without the loss of generality. For further explanation see C.4

- An objective function which has a non-zero linear term can be transformed to a purely quadratic objective function ("homogenization") by adding an extra dimension and a constraint. [68, 8, 28]
- Matrix **W** is assumed to be symmetric. For non-symmetric matrices the following transformation can be performed which changes neither the value nor the place of the global minimum. $\hat{\mathbf{W}}\quad \frac{1}{2}\ \mathbf{W}\quad \mathbf{W}^T$ [12]
- The diagonal elements of **W** are assumed to be 0, because in the case of $y_i\qquad 0\ 1$ the values of the diagonal merges into the linear term $W_{ii}y_i^2\quad W_{ii}y_i$. While in the case of $y_i\qquad 1$ they can be left out, because $W_{ii}y_i^2\quad W_{ii}\ 1$ and their sum merges into the constant term of the quadratic function.

3.2.1 Successive reduction methods for solving the UBQP

In order to develop iterative methods, we introduce a hypergraph representation of the problem and treat this material in the following order:

- First we consider the ordinary graph representation of the original problem and we put the operations of the traditional solvers into this context.
- Then we introduce a possible reduction of the original problem to smaller dimension sub-problems and give appropriate conditions for this reduction.
- The successive reductions are represented by a hypergraph and the problem solution is perceived as a path on this hypergraph.

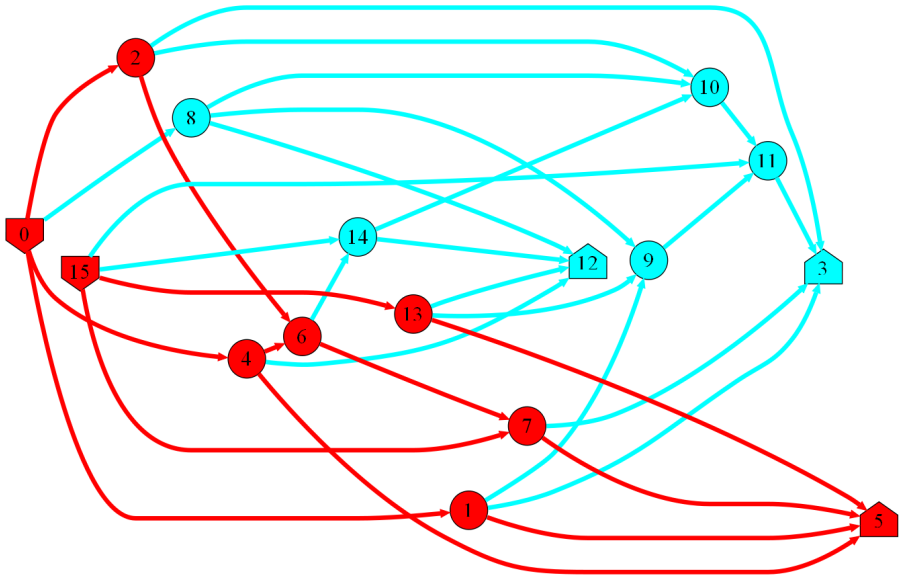
Graph representation of the UBQP The state space of the N dimension UBQP problem can be represented by a weighted graph, where the vertices of the graph correspond to the state vector **y**, the weights of the vertices are the values of the objective function $\mathbf{y}\ \mathbf{W}\ \mathbf{b}$, while the edges between the vertices are defined by a given neighborhood function. A commonly used neighborhood function defines an edge between two vertices if the corresponding two state vectors

DOI:10.15774/PPKE.ITK.2019.006

differ only in one component.

$$
\begin{aligned}
&G\ V\ E\ Q\ :\\
&V \quad \mathbf{y}\ \mathbf{y} \quad 1\ ^{N}\\
&E \quad u\ v\ \ u\ v \quad V\ \text{Neigh}\ u\ v \quad \text{TRUE}\\
&Q \quad q\ q \quad \mathbf{y}\ \mathbf{W}\ \mathbf{b} \quad \mathbf{y} \quad V
\end{aligned}
\tag{3.2}
$$

For example, Figure 24 shows the state space of a N 4 dimension UBQP represented as a weighted directed graph. The vertices are denoted by numbers 0 2^4 1 representing the decimal values of the binary state vectors. An edge is drawn here if the Hamming distance between two vertices is 1. The weights (the value of the objective function) of the vertices are noted by their left-right position, and their values can be found at the bottom in the boxes such that the larger valued vertices are at the left side, the smaller valued vertices are at the right side of the figure. All edges are directed towards a lower objective function value. In this figure we represent a vertex at a local minimum with little house shape, at a local maximum with an upside-down house shape and the transient states with circles. We highlighted with red all the vertices and edges from where the global minimum can be reached according to the given neighborhood function.



q	**y W b** :	12	6.7	5.4	4.6	4.2	3	2.2	1.7	1.3	0.97	0.95	0.67	0.62	0.5	-1.3	-1.6
	y	-1	+1	-1	+1	-1	-1	+1	+1	-1	-1	+1	+1	+1	+1	-1	-1
		-1	+1	-1	-1	+1	+1	+1	+1	-1	+1	+1	-1	-1	-1	-1	+1
		-1	+1	+1	-1	-1	+1	+1	-1	-1	+1	-1	-1	+1	+1	+1	-1
		-1	+1	-1	-1	-1	-1	-1	+1	+1	+1	-1	+1	-1	+1	+1	+1
1_to_dec	**y**	0	15	2	8	4	6	14	13	1	7	12	9	10	11	3	5

Figure 24: Search space of a 4 dimensional UBQP - vertices of a 4 dimensional hypercube

On this graph representation we can describe the key steps of well-known heuristic algorithms. Typically these are search and recombination type algorithms. For example the various forms of LS algorithms select a starting vertex in the graph and using a certain strategy - like a greedy one - search for a next vertex. They repeatedly apply this until a stopping criterion (expressed by quality

DOI:10.15774/PPKE.ITK.2019.006

or time) is not met. These heuristics are often get stuck in a vertex representing a local minimum. To avoid this, various strategy are applied. For example the SA can be interpreted as a LS, where one can escape from a local minimum by adding a noise term to the evaluation function of the greedy state transition validation. This is supposed to compensate the greediness. Recombination techniques like the GA or the MA exploit jumps in this graph. TS like heuristics try to avoid local minima by the combination of restarting from various states and remembering the visited vertices and edges leading to local minima during evaluation.

3.2.2 Breaking down UBQP into smaller dimensional sub problems

In this section I show that the the original N dimension problem can be separated into an $N \quad 1$ dimension problem and an additive term. The additive term represents the dimension to be omitted. This can be useful because if the solution of the $N \quad 1$ dimension sub problem does not depend on the omitted dimension, then this reduces the size of the state space by half. If this property holds for a sequence of dimensions, then we can apply the partitioning recursively and this way decrease the run time exponentially. Note that we refer to sub matrices and sub vectors by using a given subset of indexes.

Using the following subset of indexes: $P \quad 1 \quad N \quad 1$, we split up the original objective function into an $N \quad 1$ dimension problem and an additive term by using the notation: $\mathbf{R} \quad \mathbf{W}_{P\,P}$, $\mathbf{s} \quad \mathbf{b}_P$, $\mathbf{t}^T \quad \mathbf{W}_{N\,P}$, $\mathbf{z} \quad \mathbf{y}_P$. Note that this can be done because $W_{i\,i} \quad 0$

$$\mathbf{W} \quad \begin{matrix} \mathbf{R} & \mathbf{t} \\ \mathbf{t}^T & 0 \end{matrix} \qquad \mathbf{b} \quad \begin{matrix} \mathbf{s} \\ b_N \end{matrix}$$

$$\begin{matrix} \mathbf{y}^T\mathbf{W}\mathbf{y} & 2\mathbf{y}^T\mathbf{b} & \mathbf{z}^T\mathbf{R}\mathbf{z} & 2\mathbf{z}^T\mathbf{s} & 2y_N\ \mathbf{t}^T\ \mathbf{z} & b_N \\ \mathbf{y}\ \mathbf{W}\ \mathbf{b} & & \mathbf{z}\ \mathbf{R}\ \mathbf{s} & & g\ \mathbf{y}\ \mathbf{t}\ b_N & \end{matrix} \tag{3.3}$$

If the first term can be optimized independent of y_N then the second term needs just checking two possibilities i.e. $y_N \quad 1 \quad 1$. In this way the main challenge of minimization is going to be minimizing the first term in a reduced dimension space. Now we develop a procedure to check the condition of this partitioning.

1. We assume that the global optimum is $\mathbf{y}$. For this it is true that,

$$\mathbf{y} \quad \mathbf{W}\ \mathbf{b} \qquad \mathbf{y}^\dagger\ \mathbf{W}\ \mathbf{b} \qquad \mathbf{y}^\dagger \quad \mathbf{y} \tag{3.4}$$

2. We split up the objective functions to two components according to (3.3) and by rearranging, we get:

$$\mathbf{z} \quad \mathbf{R}\ \mathbf{s} \quad g\ \mathbf{y} \quad \mathbf{t}\ b_N \qquad \mathbf{z}^\dagger\ \mathbf{R}\ \mathbf{s} \quad g\ \mathbf{y}^\dagger\ \mathbf{t}\ b_N \tag{3.5}$$

$$g\ \mathbf{y} \quad \mathbf{t}\ b_N \quad g\ \mathbf{y}^\dagger\ \mathbf{t}\ b_N \qquad \mathbf{z}^\dagger\ \mathbf{R}\ \mathbf{s} \qquad \mathbf{z} \quad \mathbf{R}\ \mathbf{s} \tag{3.6}$$

DOI:10.15774/PPKE.ITK.2019.006

3. It is also true that $g(\mathbf{y}) - \mathbf{t}\, b_N$ is always negative, thus y_N can be computed as

$$y_N = \operatorname{sgn}(\mathbf{t}^T \mathbf{z}) - b_N \tag{3.7}$$

since $u = \operatorname{sgn}(u)\, u$, and

$$g(\mathbf{y}) - \mathbf{t}\, b_N = 2y_N(\mathbf{t}^T \mathbf{z}) - b_N = 2\operatorname{sgn}(\mathbf{t}^T \mathbf{z})(b_N - \mathbf{t}^T \mathbf{z}) - b_N = 2|\mathbf{t}^T \mathbf{z} - b_N| \geq 0 \tag{3.8}$$

Note that this indeed connects the separated dimension with the other part, because in (3.7), $\mathbf{y} = \mathbf{z}\ y_N{}^T$. Because of this we can get the optimal value for the separated dimension based on the optimum on the other part.

4. This can be used more generally to connect any point in the $N-1$ D space with a point in the ND space along one coordinate of $\mathbf{y}$. If you take an arbitrary $\mathbf{y}$, its "pair" $\mathbf{q}$ and the "N^{th} dimension reduced" version $\mathbf{p}$ will be:

$$\mathbf{y} = y_1\ y_2 \ldots y_N{}^T \quad \mathbf{p} = y_1\ y_2 \ldots 0{}^T \quad \mathbf{q} = y_1\ y_2 \ldots y_N{}^T \tag{3.9}$$

Assuming that $\mathbf{y} \in \mathbf{W}\ \mathbf{b}$ $\mathbf{q} \in \mathbf{W}\ \mathbf{b}$ then

$$\mathbf{y} \in \mathbf{W}\ \mathbf{b} \quad \mathbf{p} \in \mathbf{W}\ \mathbf{b} \quad \mathbf{q} \in \mathbf{W}\ \mathbf{b} \tag{3.10}$$

This is true since $\mathbf{W}$ has 0 diagonal elements, consequently $\mathbf{x} \in \mathbf{W}\ \mathbf{b}$ $\mathbf{x}^N$ is a linear function respect to only one x_N variable. Also note that, since $\mathbf{p} \in \mathbf{W}\ \mathbf{b}$ $\mathbf{z} \in \mathbf{R}^s$, it is generally true that

$$g(\mathbf{y}) - \mathbf{t}\, b_N \geq 0 \tag{3.11}$$

To exploit this we should have the knowledge of the optimal $\mathbf{z}$ and we should have the comparison (3.6) 2^{N-1} times which is computationally impractical.

We can develop an upper bound on the left side and a lower bound on the right side for (3.6), to use it in a search step. For example if we also assume that beside $\mathbf{y}$ in the N dimension, $\mathbf{z}$ is also optimal in the $N-1$ dimension space, the RHS of (3.6) is also non negative $\mathbf{z}^\dagger \mathbf{z}$.

One can develop another method of dimension reduction introducing a very crude bound and consequently stringent constraint, if

$$\begin{aligned} y_N &= \operatorname{sgn}(\mathbf{t}^T \mathbf{z} - b_N) \\ \max_{\mathbf{z}\ 1^{N-1}} & \left| 2(\mathbf{t}^T \mathbf{z} - b_N) \right| \leq \left| 2(\mathbf{t}^T \mathbf{z} - b_N) \right| \end{aligned} \tag{3.12}$$

then (3.6) holds $\mathbf{y}^\dagger \mathbf{y}$ and we can discard dimension N. Unfortunately this bound is too strict and in many cases dimensions are not reduced even if they could be. Analyzing numerous examples we often find that there are several dimensions that can be reduced. This leads us to introduce appropriate heuristics to perform the partitioning without the tiresome evaluation of the conditions.

DOI:10.15774/PPKE.ITK.2019.006

Hypergraph based representation To give a suitable framework for the dimension reduction we introduce the hypergraph based representation of the problem. In this the search space is extended by including all possible points from the subspaces. The extension of the search space on one hand is motivated with the reduced run time, i.e. searching for a candidate solution is performed in a smaller space, and on the other hand since we reduced the dimension we can get out from certain local minima. The new sub space points will be the intersections of the cutting plane with the edges of the hypercube.

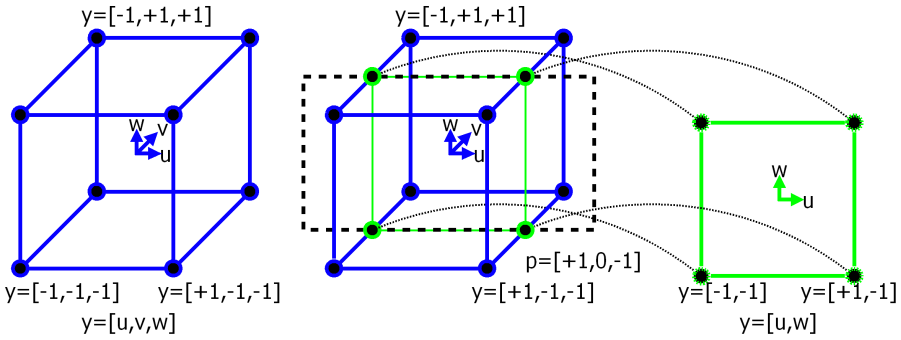


Figure 25: Discarding the 2nd dimensional from the 3 dimensional hypercube

If we do this in all possible combination we get 2^N sub spaces for search. One can connect these sub spaces with a hypergraph, where the vertices of this hypergraph are the graphs defined over the sub spaces of the original problem. In the following picture a hypergraph corresponding to the previous example is shown.

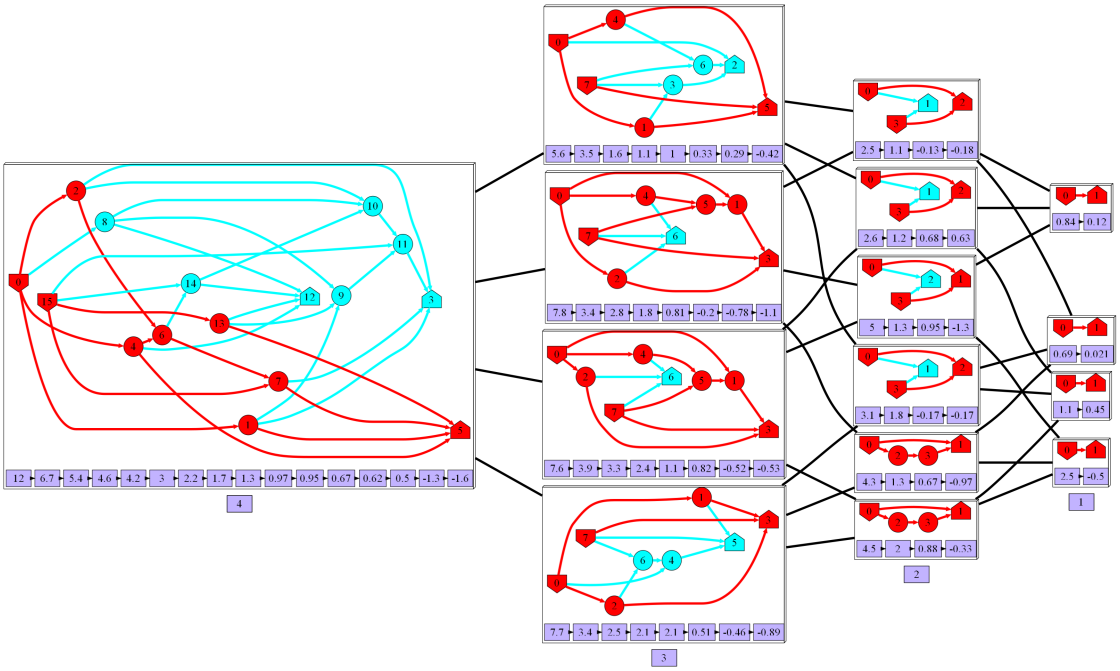


Figure 26: A hypergraph of a 4 dimensional UBQP

DOI:10.15774/PPKE.ITK.2019.006

We denote the hypergraph formally:

$$
\begin{aligned}
& G \; V_H \; E_H \\
V_H \quad & u \; u \quad G \; V \; E \; Q \; \mathbf{W}_X \; \mathbf{b}_X \quad X \\
& : \; 2^{1 \; N} \\
E_H \quad & u \; v \;\; u \; v \quad V_H \text{ s.t. } u \text{ and } v \text{ differs in 1 dimension}
\end{aligned}
\tag{3.13}
$$

3.3 Novel approaches to UBQP based on dimension reduction

In this section we present some novel hypergraph based algorithms, which will be compared to other well-known UBQP solvers. The key components of the algorithms will be shown in a flow diagram (Figure 27) and the major steps will be presented through a simple and easy-to-understand example. Furthermore a common rule set and a pseudo code will be presented which integrates all of our proposed algorithms. We introduce four algorithms with defining four variants of the rule sets. These algorithms will be used for further numerical tests. We name them "L01","D01","DA01","DA02", respectively.

First I present the flow graph representation of the algorithms:

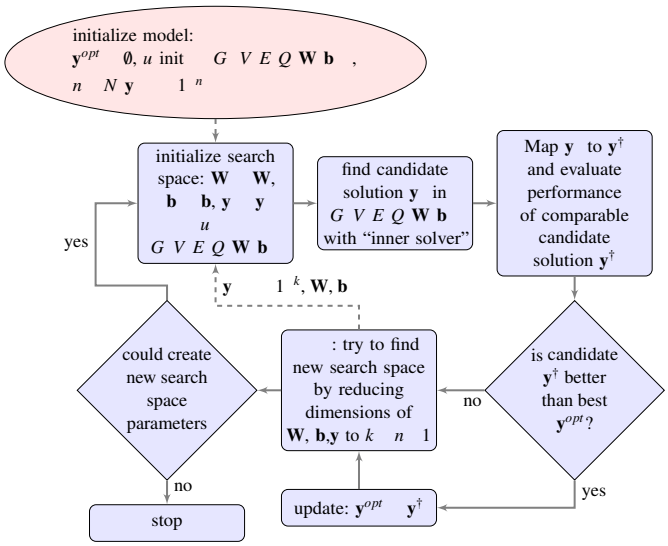


Figure 27: Flow graph representation of the algorithms

The hypergraph based algorithms operate with the following key concepts at two levels.

1. At hypergraph level: the algorithm selects an appropriate search space for the "inner solver" and traces the transition between them.
2. Within the hypergraph nodes: these hypernodes are interpreted as a corresponding UBQP. So the "inner solver" (indicated in the pseudo code later at Algorithm 4 line 17) can be any known general UBQP solver.

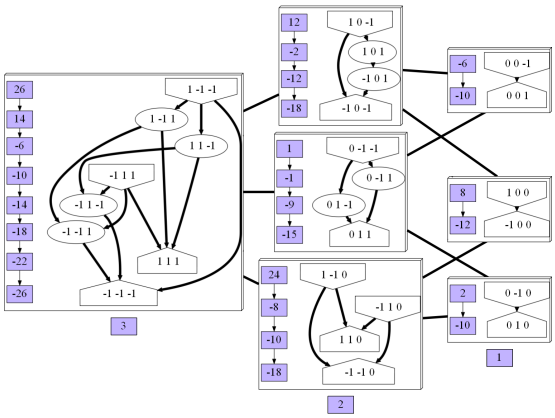
The algorithm must have at least one global storage memory where it stores the best candidate solution. This is used for comparing the iterative solutions and possibly to fine tune the strategy which selects the new hypernode. Indicated by Figure 27 we define a simple algorithm and describe each step on a 3 dimension example in order to fully understand the description of the

DOI:10.15774/PPKE.ITK.2019.006

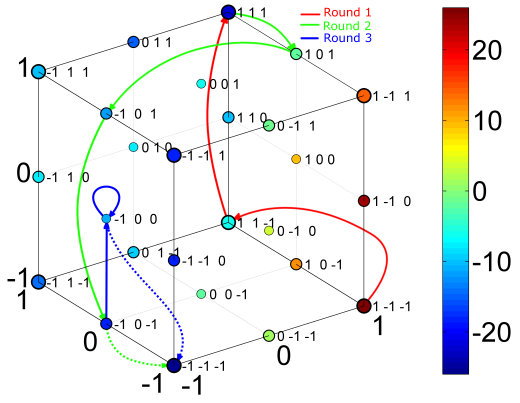
four algorithms. In this example we take the following parameters:

$$\mathbf{W}^o = \begin{matrix} -1.0 & -5.5 & -2.5 \\ -5.5 & -2.0 & -0.5 \\ -2.5 & -0.5 & -4.0 \end{matrix} \qquad \mathbf{b}^o = \begin{matrix} -5 \\ 3 \\ 1 \end{matrix}$$

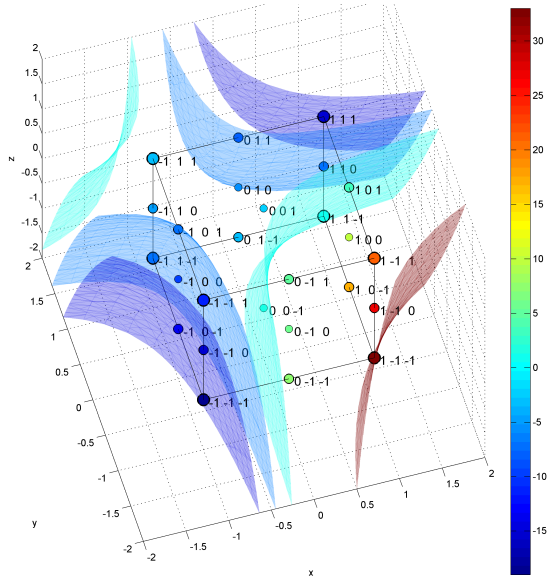
These parameters generate the hypergraph below, indicated by Figure 28a. (Note the energy values are in the boxes on the left sides). One may notice that all the lower dimension vertices in



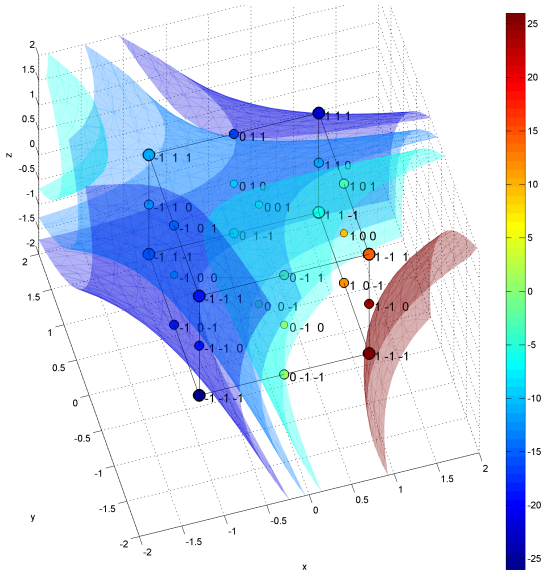
(a) Hypergraph representation



(b) Hypercube representation



(c) Sample isosurfaces of the quadratic function if $\mathbf{W}$ has 0 diagonal elements



(d) Sample isosurfaces of the quadratic function

Figure 28: Representations of a 3 dimensional problem

the hypergraph correspond to a point in the surface of the N 3 dimension hypercube. Thus the hypergraph representation (indicated by Figure 28a) corresponds to the hypercube representation (indicated by Figure 28b).

Before fully describing the algorithm the main steps are given as follows. This algorithm will start from the original N 3 dimension hypernode and perform a dimension reduction (move in the hypergraph from left to right) trying to improve its candidate solution in each step. We denote

DOI:10.15774/PPKE.ITK.2019.006

the dimension of the hypernode where the algorithm actually searches in an iteration with n . The "inner solver" can be any traditional UBQP solver. In this example we do not specify this, but we assume that this solver gives a candidate solution somehow. The full description of the algorithm is given by the following steps:

First we set the initial values of the algorithm: $\mathbf{y}^{opt}$ $\emptyset$, $\mathbf{W}$ $\mathbf{W}^o$, $\mathbf{b}$ $\mathbf{b}^o$, u init "left most graph in Figure 28a", n N 3. and we are going to store the optimal solution obtained recursively in $\mathbf{y}^{opt}$.

1: In the first round the algorithm operates on the n N 3 dimension space.
- Let us assume it starts from $\mathbf{y}$ 1 1 1^T. Then the "inner solver" searches for a candidate solution and assume it finds $\mathbf{y}$ 1 1 1^T.
- We copy $\mathbf{y}$ into $\mathbf{y}^\dagger$, the 1st solution is $\mathbf{y}^{opt}$ $\mathbf{y}^\dagger$.
- Then strategy will identify a new, lower dimension search space. For example, it discards one of the dimensions randomly. In this example let this discarded dimension be the 2nd one.

2: In the second round the search is conducted in a 2 dimensional space where new parameters are $\mathbf{W}$ $\mathbf{W}^o{}_{1\,3\ \ 1\,3}$ $\mathbf{b}$ $\mathbf{b}^o{}_{1\,3}$ u "top most graph in the center column in Figure 28a", n 2:
- Here we form a new initial state from $\mathbf{y}^{opt}$ by discarding the second component: $\mathbf{y}$ $\mathbf{y}^{opt}_{1\,3}$ 1 1^T. This corresponds to the vertex labeled by "1 0 1" in the figures.
- Now the "inner solver" searches for a new optimum and let us assume it reaches $\mathbf{y}$ 1 1^T.
- Now we copy $\mathbf{y}$ to $\mathbf{y}^\dagger$ by taking the 1st and 2nd components of $\mathbf{y}$ and placing them into the 1st and 3rd components of $\mathbf{y}^\dagger$. The value of the second component will be computed based on the values of the corresponding energy function, in this case: $\mathbf{y}^\dagger$ y_1 sgn $\mathbf{W}^o{}_{2\ \ 1\,3}$ $\mathbf{y}$ b^o_2 $y_2{}^T$ 1 1 1^T, because this will result in a lower energy.
- By comparing the energy function of the new and the previous solution we update $\mathbf{y}^{opt}$ accordingly: if $\mathbf{y^{opt}}$ $\mathbf{W^o}$ $\mathbf{b^o}$ $\mathbf{y}^\dagger$ $\mathbf{W^o}$ $\mathbf{b^o}$, then $\mathbf{y}^{opt}$ $\mathbf{y}^\dagger$; otherwise there is no change. In this example we update $\mathbf{y}^{opt}$.
- Now strategy will discard another dimension which we assume is the 3rd one.

3: In the third round the search is conducted in a 1 dimension space with parameters: $\mathbf{W}$ $\mathbf{W}^o_{1\,1}$, $\mathbf{b}$ $\mathbf{b}^o_1$, u "graph at the bottom in the right column in Figure 28a", n 1.
- The initial point of the inner solver is again a truncated version of $\mathbf{y}^{opt}$ which is $\mathbf{y}$ $\mathbf{y}^{opt}_1$ 1 which corresponds to the vertex labeled by " 1 0 0" in the figures.
- Now let us assume that the "inner solver" gives the following solution: $\mathbf{y}$ 1
- We obtain $\mathbf{y}^\dagger$ as follows: by taking the only component from $\mathbf{y}$ and placing it into the 1st component of $\mathbf{y}^\dagger$; the last discarded component will be computed based on the corresponding energy function; the rest of the discarded components (which were discarded in the previous stages of algorithm) will be copied from $\mathbf{y}^{opt}$, resulting in $\mathbf{y}^\dagger$ y_1 y_2^{opt} sgn $\mathbf{W}^o{}_{3\ \ 1}$ $\mathbf{y}$ $b^o_1{}^T$ 1 1 1^T
- By comparing the energy function of the new and the previous solution we update

DOI:10.15774/PPKE.ITK.2019.006

$\mathbf{y}^{opt}$ accordingly. In this example $\mathbf{y}^{\mathbf{opt}}\ \mathbf{W}^{\mathbf{o}}\ \mathbf{b}^{\mathbf{o}}$ $\mathbf{y}^{\dagger}\ \mathbf{W}^{\mathbf{o}}\ \mathbf{b}^{\mathbf{o}}$ so there is no update.

4-: Similarly exploring the other two 1 dimension space (the top most and the middle graph in the right column in Figure 28a) we find no better solution than the current $\mathbf{y}^{opt}$

end: Since we ran out of any further components to discard the algorithm stops and the result is $\mathbf{y}^{opt}$

It is noteworthy that if the algorithm cannot find a better solution in a lower dimension space chosen by strategy , then the algorithm returns to the end of the previous round and applies again to continue the search in a different lower dimension space.

Algorithm 4 Pseudo code of the general UBQP solver algorithm

```
1:  function INNER_SOLVER(W  k k  b  k  y init  1  k)
2:      an arbitrary UBQP minimizer
3:      return y  1  k
4:  end function
5:  function  (u  V_H  y  u_V)
6:      choose u  V_H                          choose the next hypernode and
7:      choose y  u_V                          choose a state in that hypernode
8:      return u  y
9:  end function

Input: W b and u init                          the problem and the starting hypernode
10: u  u init  V_H                             start hypernode of the alg
11: choose y  u init  V                        init state in the hypernode
12: repeat
13:     define L W b y  objective function
14:     u  u and y  y
15:     W b  parameters from u  G V E Q W b
16:     if SHOULD_EMPLOY_INNER_SOLVER( )then
17:         y  INNER_SOLVER(W b y)
18:     else
19:         y  y
20:     end if
21:     u y  (u y )
22: until STOP_CRIT( )
Output: y                                      the best solution found by the alg.
```

Based on this intuitive example, one may specify the rules of the general algorithm as follows:

1. Specify the starting hyper node of the algorithm. u init ? u init V_H
2. Choose an "inner solver" (Algorithm 4 line 1) that we apply in a hypernode to obtain the solution of the corresponding UBQP defined over this hypernode.
3. The performance of an n dimension candidate solution is defined by the corresponding value of the energy function over this candidate solution.
4. Specify a strategy which selects the next hypernode (Algorithm 4 line 21).
5. Select an initial state for the "inner solver" in the chosen hypernode.
6. Give the overall stopping criterion of the algorithm. (Algorithm 4 line 22)

DOI:10.15774/PPKE.ITK.2019.006

Although the general algorithm enables us to define its several variants we present only four, named as: “L01”,”D01”,”DA01”,”DA02”, respectively. These four algorithms are characterized by two major properties: (i) selecting a better solution by checking all possibilities; or selecting the first better solution with respect to the search criterion. One may also differentiate among the algorithms based on (ii) reducing or extending the dimensions. Table 3 demonstrates the possibilities.

Table 3 Categorization of the algorithms

	greedy	opportunistic
dim. reducer	L01	D01
dim. adder	DA02	DA01

Here we shortly describe the major properties of the four algorithms. (The precise description of the algorithms can be found in Appendix F.)

L01 starts from the original N dimension space, and iteratively reduces the dimension in a greedy fashion. It searches all reachable n 1 dimension hypernode from a given n dimension hypernode, and selects the best one.

D01 also iteratively reduces a dimension, but does not search all reachable n 1 dimension hypernodes, but if it finds a promising one, then it chooses that hypernode.

DA01 builds up the solution by increasing the dimension instead of reducing it. It starts from a 1 dimension hypernode and iteratively increases the dimension one-by-one on its candidate solution. This algorithm chooses the next hypernode by an opportunistic manner: if the chosen hypernode looks promising from the point of energy, it selects that.

DA02 is also an algorithm which increases the dimensions. It differs from “DA01” only in the choice of the next hypernodes. The algorithm examines all reachable n 1 dimension hypernodes from an n dimension one and selects the best among them according to energy.

THESIS II.1 (A heuristic solver family based on hypergraphs for UBQP)**.** *In Algorithm 4, I have given a hypergraph based, easily parallelizable algorithm family to sub-optimally solve the UBQP problem. The algorithms project the original search space into a hypergraph representation and use a HNN based internal solver to find a solution. I have given four instances of which two employs dimension reduction and two dimension addition. Table 3 summarizes the operation modes of the instances. (The precise description of the algorithms can be found in Appendix F) I have tested the performance on three different problem sets: on the standard ORLIB UBQP benchmark set (subsection 3.4), on a scheduling problem (subsection 3.5), and on a simulated MUD problem (subsection 3.6). I have shown that the proposed methods perform near optimal on the investigated ICT problems.*

The thesis is restated in a self consistent way in Appendix A at Thesis II.1 (page 87).

3.4 Performance analysis of the novel algorithms

In this section we present a numerical performance analysis of our proposed algorithms and compare them with some well-known UBQP solvers. We took the problems from the ORLIB [10] test set. We have implemented all the algorithms in the same programming environment

DOI:10.15774/PPKE.ITK.2019.006

and performed the tests within the same computational environment as well. First we present the numerical performance on a few specific test problems. Then we give the overview of the results for all the defined test problems in a table. Besides the algorithms presented in subsection 3.3 we also use the following traditional algorithms for the sake of comparison:

- "HNN" - a discrete time and discrete valued Hopfield type recurrent network.
- "1-opt LS" - a 1-opt local search type algorithm.
- "BLS" an algorithm presented by Beasley [11] based on a 1-opt local search method.
- "BTS" a taboo search algorithm also presented by Beasley [11].
- "DDT" the well-known DDT algorithm by Boros, Hammer, and Sun [14]
- "SDR" an SDR type algorithm without randomization presented by Luo et al. [108].
- "SDR with randomization" is the same algorithm but with randomization [108].

All algorithms were reimplemented in Matlab by the author reusing pre-existing Mathworks provided libraries to get comparable execution times. For the comparison of the execution times, the author assumed that the used libraries are fair and scale linearly. They are fair in the sense that by using two different components with similar computational complexity they are executed approximately in the same time. It also worth noting that all simulations were performed on a sequential architecture without any parallelization.

The algorithms can be divided into two major groups:

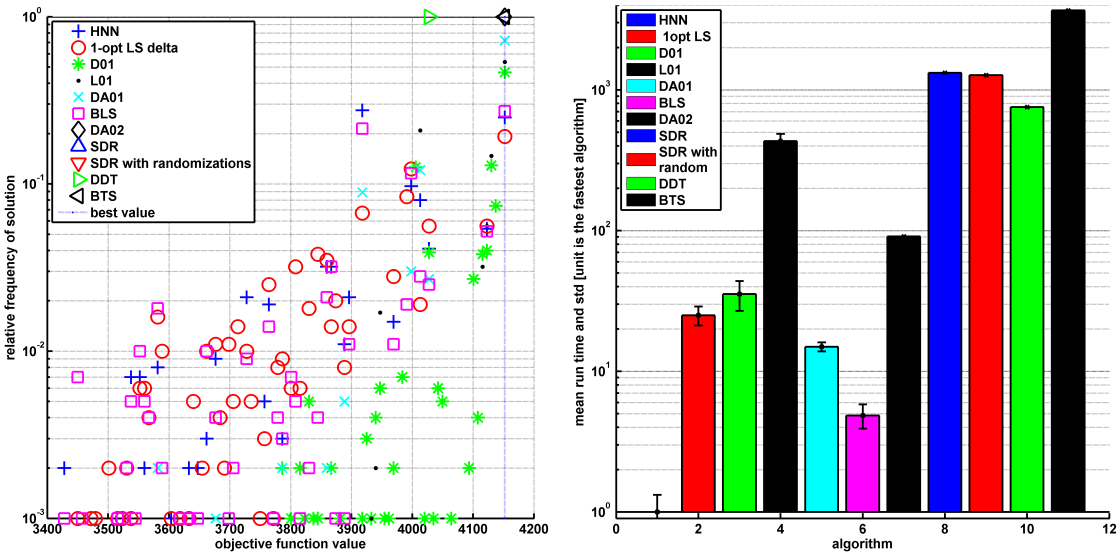
- the deterministic algorithms, such as DA02, SDR, BTS, DDT
- stochastic algorithms, such as HNN, D01, L01, DA01, BLS.

The performance measure is a histogram taken over the solutions computed by re-running the different algorithms 1000 times. In the figures the markers represent the top of the corresponding histogram bins. In the ORLIB the variable $\mathbf{y}$ of the UBQP problems have $\mathbf{y}$ 0 1 ^{N} valued components, and the objective function is a maximization problem, although an equivalent minimization problem with $\mathbf{y}$ 1 ^{N} can be obtained, we choose to present the numerical values like the ORLIB defines it for the sake of further comparability by other authors. The figure belowFigure 29 shows the performance of the algorithms on the 5^{th} problem (selected from the set of problems with 50 dimension). From this figure it can be seen that all the BTS, DA02, DA01, L01, D01, BLS, HNN, 1opt LS algorithms find the global maximum. The DDT and both variations of the SDR perform poorly on this problem. The ordering of the probabilistic algorithms based on the relative frequency of finding the global maximum is the following: DA01, L01, D01, BLS, HNN, 1opt LS. The BLS and the simple HNN performs almost identically. Figure 29b presents the relative run times of the algorithms. The unit time was chosen to be the run time of the fastest algorithm (HNN).

Figure 30a depicts the performance of the algorithms on the 7^{th} problem (selected from the set of problems with 100 dimension). In this case the histogram is obtained based on 100 independent runs.

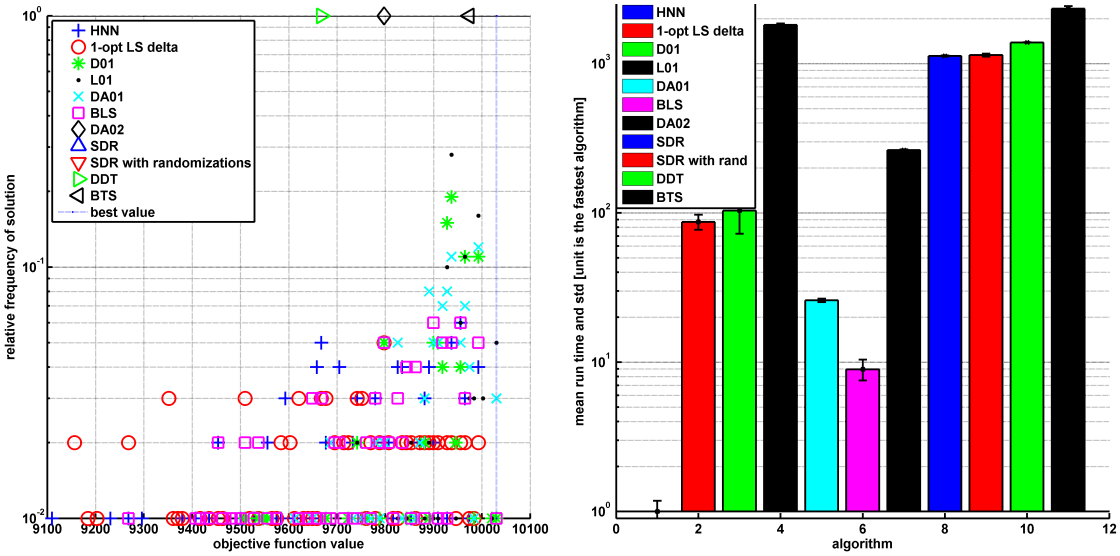
It can be seen that L01, DA01, BLS and D01 algorithms are capable of finding the best solution. The BTS every time found the same and fairly good solution due to its deterministic nature. From the histogram it can be seen that algorithms D01, L01 and DA01 perform as good as the BTS in

DOI:10.15774/PPKE.ITK.2019.006



(a) Relative frequencies of the reached solutions (b) Relative run times of the algorithms

Figure 29: Performance on the 5th problem from the ORLIB 50 dimensional problems.



(a) Relative frequencies of the reached solutions (b) Relative run times of the algorithms

Figure 30: Performance on the 7th problem from the ORLIB 100 dimensional problems.

average, however sometimes they yield a better solution. The SDR solvers perform poorly on this problem. Comparing the relative run times on this problem, it can be seen that the BTS proves to be the slowest one.

The next figures show the performance on the 2nd problem from the 500 dimension problem set. In this case we only present the histogram based on 50 independent runs. In this case the best solution was given by the BTS, L01 and D01 algorithms. From the run time analysis it can be seen that on this problem the BTS only runs 2 times slower than the D01 algorithm, but the BTS always finds a good quality solution as opposed to the D01. Based on the ranking obtained by the run-time analysis, it can be concluded that most of the time our proposed algorithms perform better or identical compared to algorithms of similar complexity.

DOI:10.15774/PPKE.ITK.2019.006

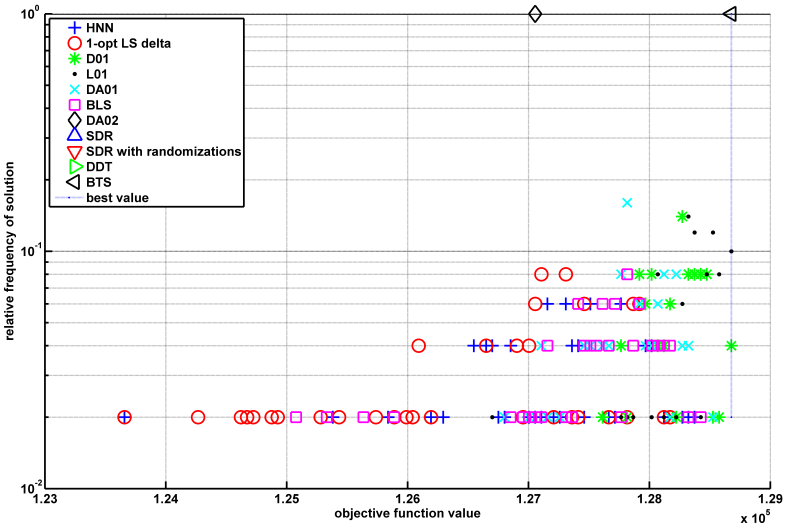


Figure 31: Performance on the 2nd problem from the ORLIB 500 dimensional problems.

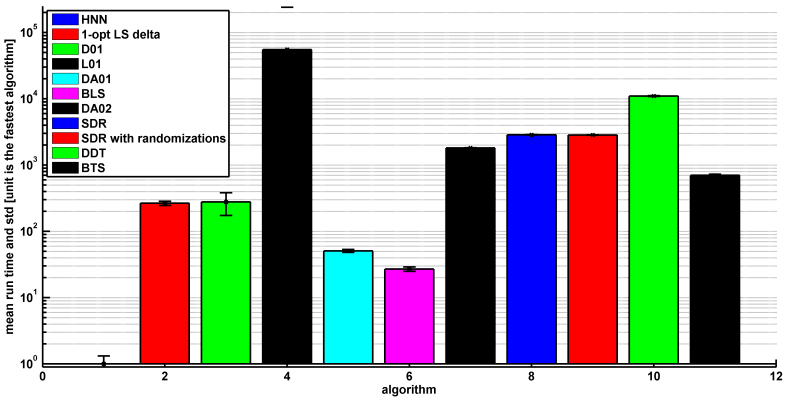


Figure 32: Relative run times on the 2nd problem from the ORLIB 500 dimensional problems.

We must note that in the worst case HNN uses the same amount of time as the greedy local search. This happens rarely but their performances are almost identical. Regarding speed, HNN can be matched with the DA01 algorithm, which builds up a solution in a bottom-up manner, in one pass. On the other hand, DA01 always perform better than HNN. We must also note that the proposed algorithms can be easily parallelized: The strategy defines a tree and evaluating the performances of the branches of the tree are independent processes, therefore they can be done in a parallel manner. This can reduce the run time or may boost the performance of the solution if an appropriate architecture is chosen. The tables in the Appendix F.5 present the performance and the run time analysis for all the problems.

Comparison of 3 algorithms

To make a more profound comparison, we present figures depicting only the performance of the HNN, the DA01 and the BTS algorithms on a larger run set. In the following figure a histogram of the performance can be seen based on 10,000 runs on a 50 and a 100 dimension problem, respectively. Furthermore, the performance on a 500 dimension problem were tested by 1000 runs. Table 4 summarizes the performance and run time values for the selected algorithms.

DOI:10.15774/PPKE.ITK.2019.006

It can be seen that the proposed DA01 algorithm performs better for the lower dimension problems than BTS if the run time is the same. On the other hand the DA01 reaches the best solution in a shorter time for lower dimension problems.

We must note that in this comparison we run the DA01 algorithm in each iteration independently and the best result has been selected as the final result. This is a very naive approach, but this lends itself to easy parallel implementation.

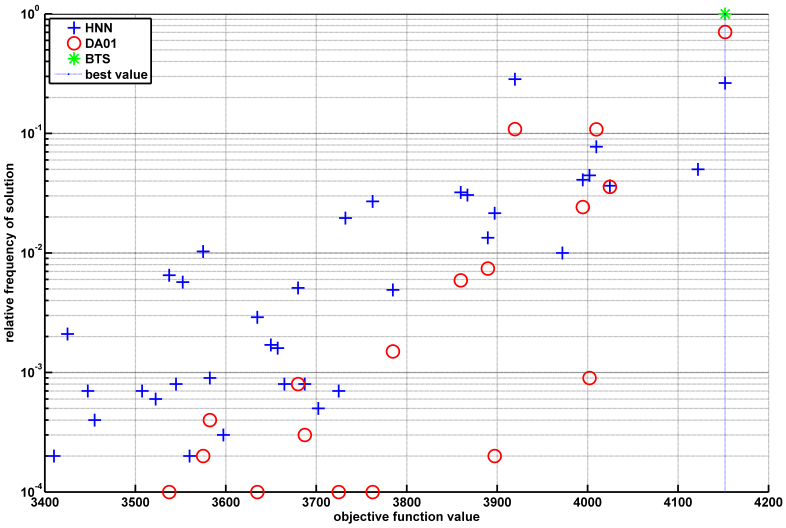


Figure 33: Performance on the 5th of the 50 dimensional problems.

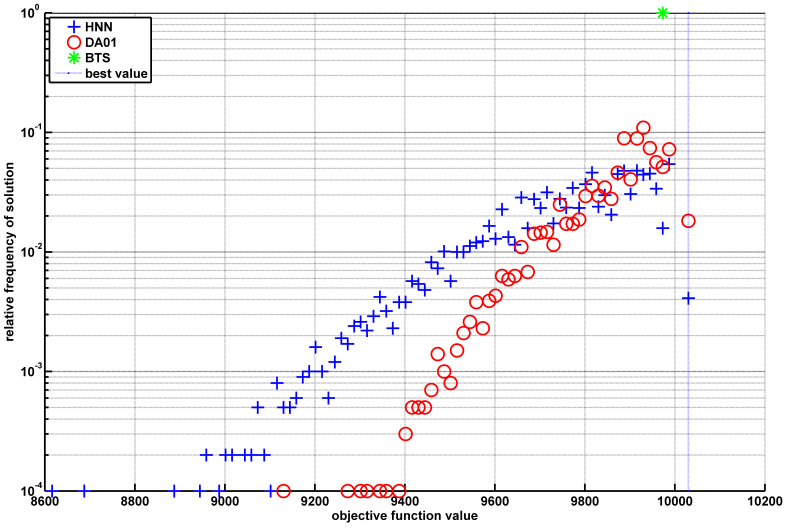


Figure 34: Performance on the 7th of the 100 dimensional problems.

DOI:10.15774/PPKE.ITK.2019.006

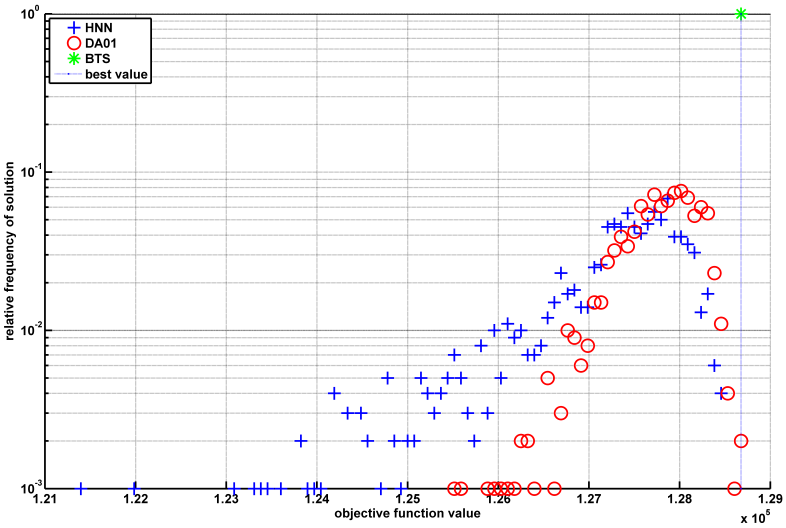


Figure 35: Performance on the 2nd of the 500 dimensional problems.

Table 4 Performance comparison of BTS vs DA01

		solution		best solution		relative freq of best solution		mean run time		mean run time until best solution found	
prob	alg	BTS	DA01	BTS	DA01	BTS	DA01	BTS	DA01	BTS	DA01
dim=50 K=10000	1	***2160***	2055,4	***2160***	***2160***	1	0,1052	232,23	1	232,2286	***9,5057***
	2	***3658***	3588,6	***3658***	***3658***	1	0,1316	251,05	1	251,0545	***7,59878***
	3	4650	***4680,4***	4650	***4778***	1	0,4924	240,36	1	240,3645	***2,03087***
	4	***3472***	3400,7	***3472***	***3472***	1	0,1275	234,53	1	234,5311	***7,84314***
	5	***4152***	4098,5	***4152***	***4152***	1	0,7053	234,99	1	234,9933	***1,41784***
	6	***3842***	3823,6	***3842***	***3842***	1	0,5405	239,71	1	239,7059	***1,85014***
	7	***4588***	4535,1	***4588***	***4588***	1	0,2746	241,11	1	241,107	***3,64166***
	8	***4222***	4195,2	***4222***	***4222***	1	0,8435	233,71	1	233,7129	***1,18554***
	9	***3862***	3829,9	***3862***	***3862***	1	0,6273	228,99	1	228,9912	***1,59413***
	10	***3496***	3450,8	***3496***	***3496***	1	0,2163	236,26	1	236,264	***4,62321***
prob	alg	BTS	DA01	BTS	DA01	BTS	DA01	BTS	DA01	BTS	DA01
dim=100 K=10000	1	***7910***	7631	***7910***	***7910***	1	0,0039	65,814	1	***65,8142***	256,41
	2	***11178***	11030	***11178***	***11178***	1	0,0192	75,619	1	75,61899	***52,0833***
	3	***12956***	12875	***12956***	***12956***	1	0,1854	73,601	1	73,60066	***5,39374***
	4	***10606***	10493	***10606***	***10606***	1	0,0722	70,157	1	70,15745	***13,8504***
	5	***8994***	8777,2	8994	***8996***	1	0,0173	70,659	1	70,6588	***57,8035***
	6	***10470***	10362	10470	***10486***	1	0,011	71,521	1	***71,5213***	90,9091
	7	***9980***	9877,6	9980	***10030***	1	0,0182	72,419	1	72,41868	***54,9451***
	8	***11380***	11240	***11380***	***11380***	1	0,1357	71,742	1	71,74218	***7,3692***
	9	***11340***	11246	***11340***	***11340***	1	0,269	79,918	1	79,91803	***3,71747***
	10	***12438***	12348	***12438***	***12438***	1	0,032	79,959	1	79,95852	***31,25***
prob	alg	BTS	DA01	BTS	DA01	BTS	DA01	BTS	DA01	BTS	DA01
dim=500 K=1000	1	***116526***	114780	116526	***116532***	1	0,001	13,642	1	***13,6418***	1000
	2	***128678***	127801	***128678***	***128678***	1	0,002	14,043	1	***14,0425***	500
	3	***131084***	130013	***131084***	***131084***	1	0,006	13,294	1	***13,2944***	166,667
	4	***129794***	128646	***129794***	129784	1	0,001	13,209	1	***13,2092***	1000
	5	***125008***	123859	125008	***125062***	1	0,001	11,694	1	***11,6937***	1000
	6	***121868***	120189	***121868***	***121868***	1	0,001	13,336	1	***13,3355***	1000
	7	***122730***	121163	122730	***122756***	1	0,001	13,989	1	***13,989***	1000
	8	***123454***	121958	***123454***	123428	1	0,001	13,811	1	***13,8106***	1000
	9	***121622***	120026	121622	***121668***	1	0,001	13,889	1	***13,8885***	1000
	10	***130900***	130219	130900	***131374***	1	0,01	13,865	1	***13,8647***	100

Parallelization possibilities

The following section demonstrates on a simple numerical example, that the introduced algorithms lend themselves to parallelization. I took the "L01" greedy (and slowest) algorithm and instead of executing the "inner solver" part in a sequential manner (by a simple "for loop" over all possible sub-hypernodes), I used the Matlab provided parallel-for "parfor" type of iteration, thus the parallelized version differs from the original by only one line. The solvers were compiled by

the Matlab coder into a mex binary form, thus using the OpenMP application interface to generate multicore code when the “parfor” was used. The following figure depicts the “L01” algorithm for 5 variants: L01,L01_p1,L01_p2,L01_p3,L01_p4. L01 is the original unmodified solver, L01_p1 if the parfor enabled algorithm but the number of execution threads were limited to 1. L01_p2 uses 2, L01_p3 uses 3 and L01_p4 uses 4 threads. At the time of writing I only had access to a machine which had 4 physical cores, so I did not scale the algorithm further. The test were executed on an Intel(R) Core(TM) i7-8650U CPU @ 1.90GHz based machine. Since the algorithms are identical, the found solutions are identical as well, thus those results are not depicted for the sake of brevity. The following figure depicts the mean runtimes and the standard deviations of the corresponding algorithm variants. As expected it can be seen that L01_p4 executes the fastest, although the

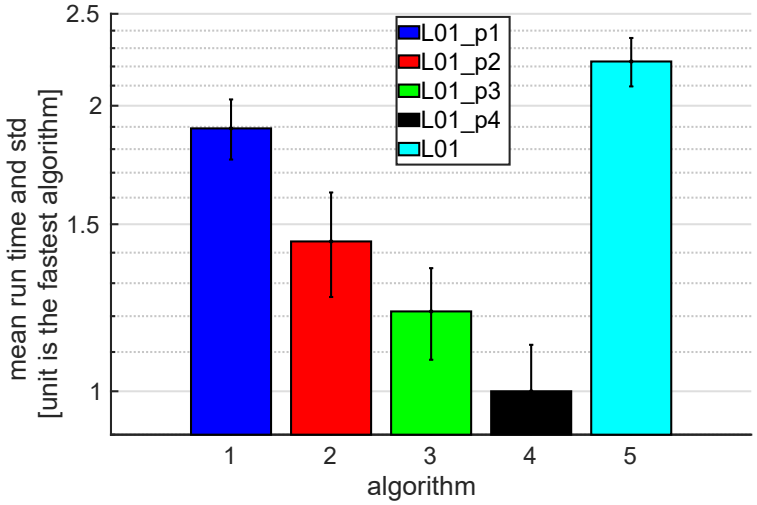


Figure 36: Execution time comparison of the L01 algorithm with parallelization.

speedup does not scale by the number of threads used. L01_p1 is 1.9 times slower than L01_p4, but the overall speedup scales linearly in this domain. This naiive implementation is just a simple demonstration that the algorithms are easily parallelizable and does not show the full gain to be earned by parallelization therefore are treated as a lower bound on the possible speedup gain. It is worth to mention that even using a parfor loop with one execution thread performs better than the simple sequential one, which is presumably because of the efficiency of the OpenMP inteface and the underlying code generation mechanism. It is also worth mentioning that the proposed algorithms have multiple independently executable algorithmic parts, therefore lend themselves to massively parallel implementations, e.g on GPU or on FPGA.

3.5 Application of UBQP to Scheduling

As was mentioned in section 1, scheduling plays a central role in communication technologies nowadays. It is widely applied in IoT, cloud computing for load balancing and task distribution, in buffered packet switching systems for call admission control [4] or in various problems in the field of WSN such as the scheduling of TDMA communication in clustered WSN protocols. The efficient collection of data from multiple parties [104, 23, 101] can be also regarded as a scheduling problem [38]:

DOI:10.15774/PPKE.ITK.2019.006

In our model there are J jobs to be allocated to a capacity constrained resource. The amounts of jobs are expressed in $X_j \ j \quad 1 \quad J$ units. We assume that the jobs can be divided into discrete time slots, and in these time slots the jobs are considered to be preemptive. The resource capacity is limited denoted by C, meaning that the resource can consume C units of job in a time instant. Each job $X_j \ j \quad 1 \quad J$ has a deadline $K_j \ j \quad 1 \quad J$ and within this time the job is supposed to be completed. The jobs are also given priorities expressed by weight vector $\mathbf{w} \quad w_1 \ w_2 \quad w_J \ w_j$, the larger the weight the more important the job is. The schedule of job j can be represented as a binary vector $\mathbf{s} \ j \quad 0 \ 1 \ ^{K_j}$, if $s_t \ j \quad 1$ then a unit of X_j is scheduled to be served by the resource at time instant t. A schedule of the individual jobs $j \quad 1 \quad J$ build up a scheduling matrix $\mathbf{S}$ where $S_{j\,i} \quad \begin{matrix} s_i \ j & i & K_j \\ 0 & i & K_j \end{matrix}$, $L \quad \max_j K_j, \ i \quad 1 \quad L$. The following conditions will guarantee a valid scheduling matrix: Summing the scheduling matrix row wise $\sum_{t \ 1}^{L} \mathbf{S}_{j\,t} \quad X_j \quad j$, one can check the amount of time slots scheduled for a job, and summing the matrix column wise $\sum_{j \ 1}^{J} \mathbf{S}_{j\,t} \quad C \quad t$, one can see how many job units are scheduled at a given time instant.

For example if $C \quad 2, J \quad 4, X_1 \quad 5, X_2 \quad 4, X_3 \quad 3, X_4 \quad 7 \ K_1 \quad 10, K_2 \quad 8 \ K_3 \quad 4$ and $K_4 \quad 9$, a valid scheduling matrix could be:

$$\mathbf{S} \quad 0 \ 1 \ ^{J \ L} \quad \mathbf{S} \quad \begin{matrix} 0 & 0 & 1 & 0 & 1 & 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 1 & 0 & 1 & 1 & 1 & 0 & 0 \\ 1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 0 & 1 & 1 & 1 & 0 & 1 & 0 \end{matrix}$$

The scheduling matrix can be invalid for two reasons:

- more or less time slots are assigned to a job: j s.t. $\sum_{t \ 1}^{L} \mathbf{S}_{j\,t} \quad X_j$
- the resource is overflowed: t s.t. $\sum_{j \ 1}^{J} \mathbf{S}_{j\,t} \quad C$

As a result, we perceive the scheduling problem as a constrained optimization, where we want to minimize the weighted tardiness which is defined below, given that the capacity constraint is not violated at any time instant. The tardiness of job j is defined as the remaining uncompleted job after completing the schedule, which can be formalized as follows:
The ending of job j in a particular scheduling $\mathbf{S}$ is denoted by $F_j \quad \text{argmax}_{t \ 1 \ L} S_{j\,t} \quad 1$, the last 1 in each row of $\mathbf{S}$. Then the corresponding tardiness is defined as: $T_j \quad \max \ 0 \ F_j \quad K_j$ (the number of unfinished units of X_j after the completion of the schedule). This gives rise to the formal definition of the constrained optimization problem

$$\begin{aligned} \mathbf{S}^{opt} \ : \ & \underset{\mathbf{S} \quad 0 \ 1 \ ^{J \ L}}{\text{argmin}} \ \sum_{j \ 1}^{J} w_j T_j \\ \text{s.t.} \ & \sum_{t \ 1}^{L} \mathbf{S}_{j\,t} \quad X_j \quad j \\ & \sum_{j \ 1}^{J} \mathbf{S}_{j\,t} \quad C \quad t \end{aligned} \tag{3.14}$$

In order to map the constrained optimization problem into an unconstrained one, we use the

DOI:10.15774/PPKE.ITK.2019.006

method of Kochenberger [91] to incorporate the constraints into a quadratic form by adding them to the objective function as linear terms.

$$\mathbf{S}^{opt} : \underset{\mathbf{S}\ 0\,1^{J\ L}}{\operatorname{argmin}} \sum_{j\ 1}^{J} w_j T_j + \sum_{j\ 1}^{J}\left(\sum_{t\ 1}^{L} \mathbf{S}_{j\,t} - X_j\right)^2 + \sum_{t\ 1}^{L}\left(\sum_{j\ 1}^{J} \mathbf{S}_{j\,t} - C\right)^2 \tag{3.15}$$

In this way, the scheduling problem can represented by a UBQP problem.

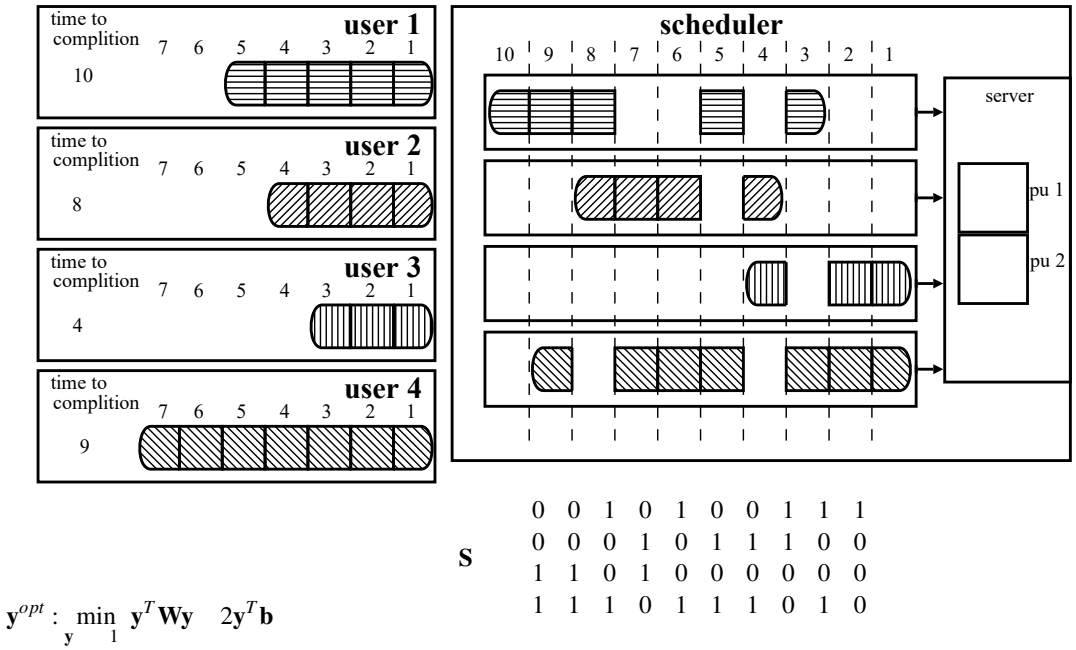


Figure 37: Visualization of the scheduling problem

$$\sum_{j\ 1}^{J} w_j T_j \quad \mathbf{y}^T\mathbf{W}^{A}\mathbf{y} \quad 2\mathbf{b}^{A\ T}\mathbf{y} \tag{3.16}$$

$$\sum_{j\ 1}^{J}\left(\sum_{t\ 1}^{L} \mathbf{S}_{j\,t} - X_j\right)^2 \quad \mathbf{y}^T\mathbf{W}^{B}\mathbf{y} \quad 2\mathbf{b}^{B\ T}\mathbf{y} \tag{3.17}$$

$$\sum_{t\ 1}^{L}\left(\sum_{j\ 1}^{J} \mathbf{S}_{j\,t} - C\right)^2 \quad \mathbf{y}^T\mathbf{W}^{C}\mathbf{y} \quad 2\mathbf{b}^{C\ T}\mathbf{y} \tag{3.18}$$

The strategy for setting the heuristic parameters and the details of the transformation to the quadratic form can be found in [38].

Performance analysis for scheduling

Here we present some numerical results for solving the scheduling problem by the proposed methods developed for UBQP. The simulation has been carried out in a similar way as presented

DOI:10.15774/PPKE.ITK.2019.006

in [38]. Namely, the jobs X_i i 1 J have been generated between 1 and 10 subject to discrete uniform distribution, similarly K_i i 1 J have been generated between X_i and X_i 5, and the value of w_i i 1 J have been taken from the range 1 10 . The capacity of the server was set as C J 4.

First we show the value of TWT with respect to the different choice of parameter . Because we use the same method as in [38], we run an exhaustive search for different values of parameter and select the best solution. For each different choice of , every algorithm started from a a randomly chosen initial state and was run several times. Their best performance is shown on the next figures with solid lines. Referring to the performance analysis Figure 33 in subsection 3.4 dedicated to ORLIB, these curves correspond to the right edges of the histograms. The flatness of the curves indicate how resilient an algorithm is to the choice of parameter . We have selected the best solution from these values.

The next figure depicts the results of the simulation runs obtained with the following parameters: J 10 and **X** 9 10 2 10 7 1 3 6 10 10 , **K** 10 15 7 12 11 2 5 11 14 15 , **w** 7 1 9 10 7 8 8 4 7 2 , C 3

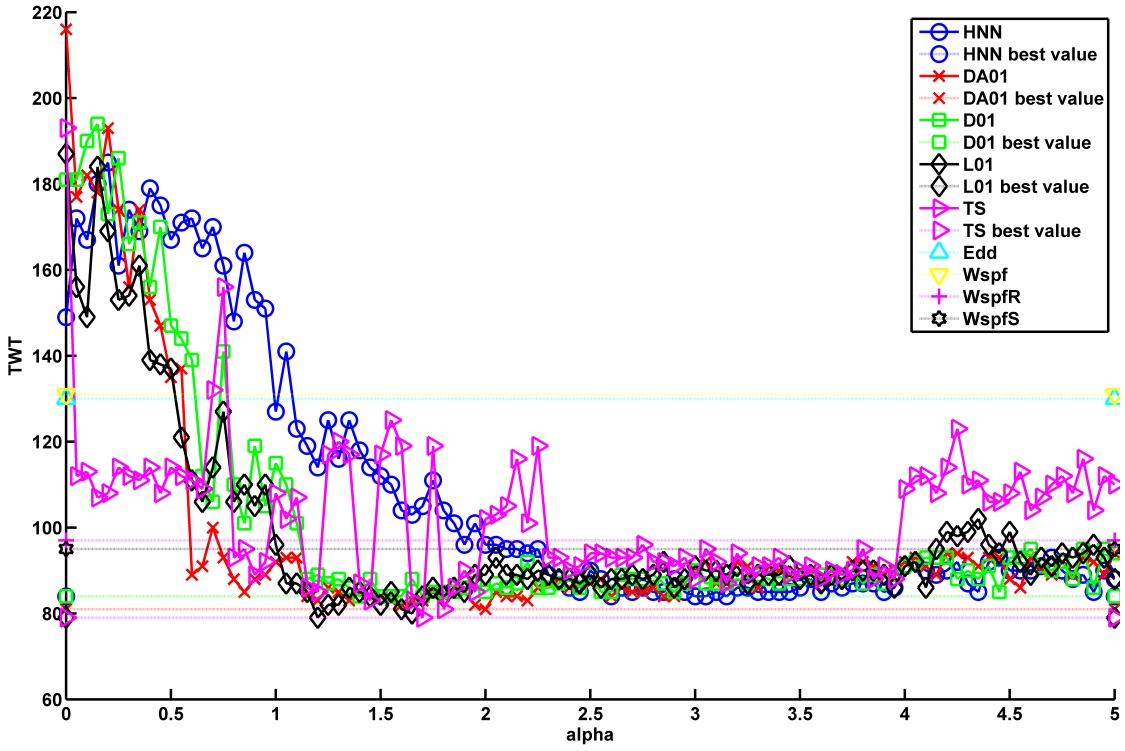


Figure 38: TWT performance of algorithms versus heuristic parameter for a specific case J 10

In this case the best TWT values achieved by the different algorithms are: HNN:84, DA01:81, D01:84, L01:79, TS:79, EDD:130, WSPT:131 WSPTR:97, WSPTS:95, respectively. For this specific case, L01 and Taboo Search perform the best, but their performances are nearly identical with the others. The best TWT values can be found when is in the range of 1 2 3 . The next figure (Figure 39) shows the difference between the values of the objective function of the UBQP and the values achieved by the DHNN solver. The curve depicting the difference of the best found solutions are indicated with a solid line. If the curve is above 0 then the corresponding

DOI:10.15774/PPKE.ITK.2019.006

algorithm yields a better solution than the DHNN with respect to the quadratic function value.

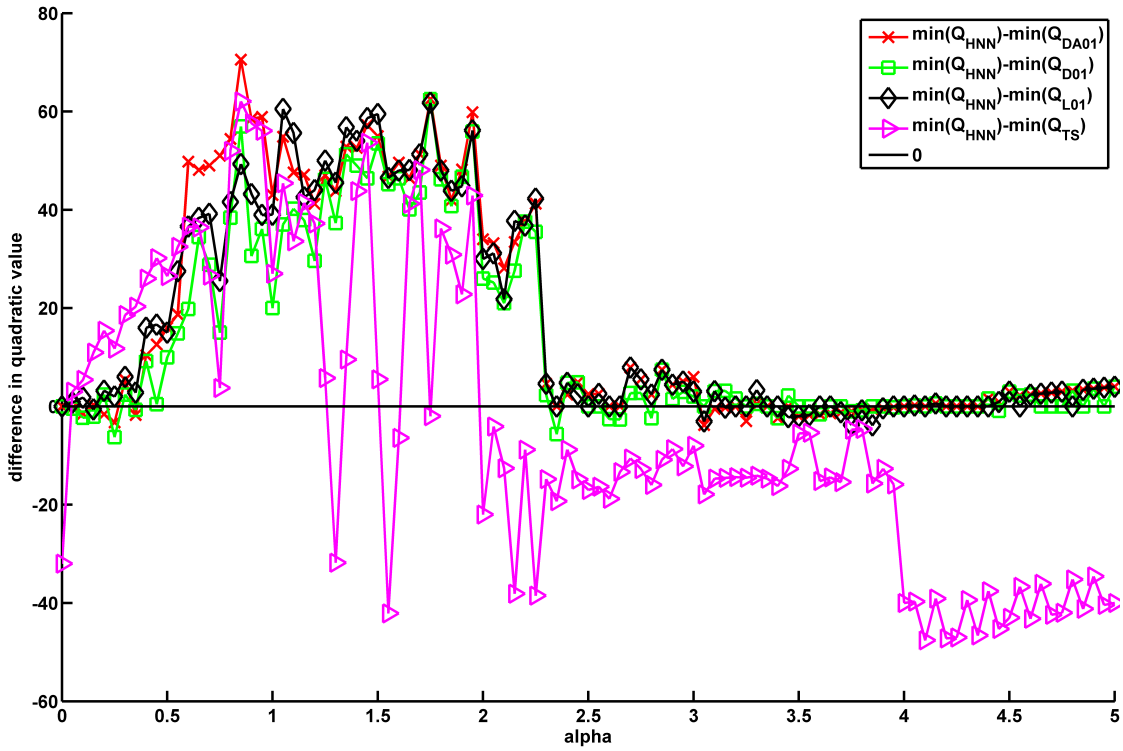


Figure 39: Quadratic value performance of algorithms respect to the performance of HNN versus heuristic parameter for a specific case J 10

It can be seen by comparing Figure 39 and Figure 38 that parameter exceeding the value 2 3, all the quadratic solvers (except for TS) find a solution which are of roughly the same value, but they sometimes yield different TWTs. This is due to the reparation effect, because if we find a solution which is not a valid scheduling matrix, we use the reparation method used in [38]. After parameter exceeding the value 2 3 the likelihood of finding a solution which does not satisfy the two required constraints is growing steadily. In the region of 0 5 2 3 the proposed algorithms outperform the basic HNN as well as TS methods. It can be shown that the new methods usually find better solutions than the plain DHNN solver, and also have flatter TWT curves. This implies that in these cases the term corresponding to the quality of solution can dominate the terms corresponding to the constraints in the objective function (3.15). As a result these methods may yield better solutions.

To summarize the average performance of the algorithms the next figure indicates the average TWT with respect to the number of users (The average TWT value was calculated over 100 sample set)

Note that for J 15 the TS and L01 algorithm would require unreasonably high run times so they were not analyzed here. From this figure it can be seen that the DA01 algorithm is the clear winner, although the other proposed algorithms also perform well as opposed to the traditional scheduling algorithms.

DOI:10.15774/PPKE.ITK.2019.006

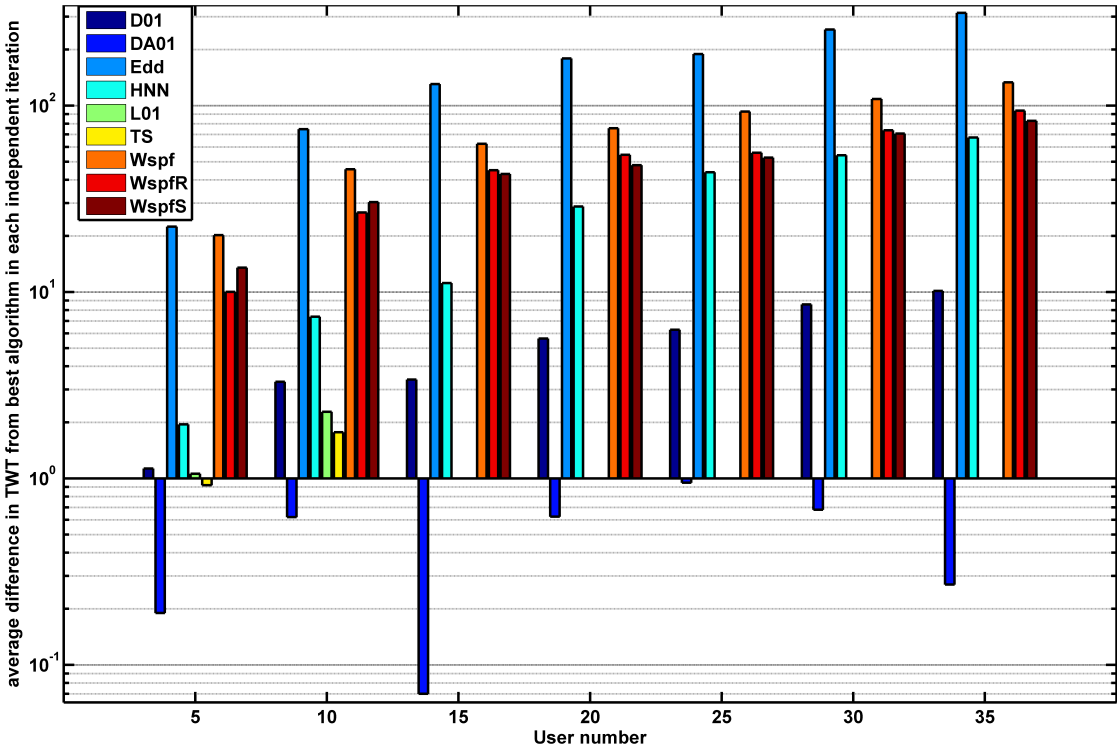


Figure 40: Relative average TWT from best solution in each iteration

3.6 Application of UBQP to MUD in an environment with fading

Binary pattern recognition plays a central role in modern digital communication systems. This task is especially important in wireless communication technologies when binary messages corrupted by fading and noise are to be restored. The PHY using radio as a media has to account for phenomena like scattering, multi path propagation and simultaneous presence of multiple radio waves. For efficient use of radio spectrum and transmission energy, one of the most frequently used access mode in modern wireless communication technologies is CDMA. In case of CDMA the received sequence is subject to MUI combined with time delays and ISI resulting from "non orthogonal code words" and from the channel distortion, respectively.

A multi user CDMA system of M 1 scenario (for example in the uplink direction) can be represented [33, 34] by the following block diagram in Figure 41. Here $\mathbf{y}$ 1 ^{L} is the transmitted binary sequence and there are M users to communicate simultaneously in K length blocks (L K M). $\mathbf{H}$ is a matrix of L L which represents the channel distortion resulted by the properties of the radio propagation. The specific element of $\mathbf{H}$ can be computed if the impulse responses of the channel and the spread codes are known [33, 34, 162]. The additional noise subject to multi dimensional normal distribution comes from the interference between the codes used by the system and by the interference of other radio communication that does not belong to this system:

$$0 \qquad N_0\mathbf{C} \ \mathbf{C}^T \tag{3.19}$$

where N_0 is the ambient noise power and $\mathbf{C}$ contains the spread codes of the users row wise and $\mathbf{x}$

DOI:10.15774/PPKE.ITK.2019.006

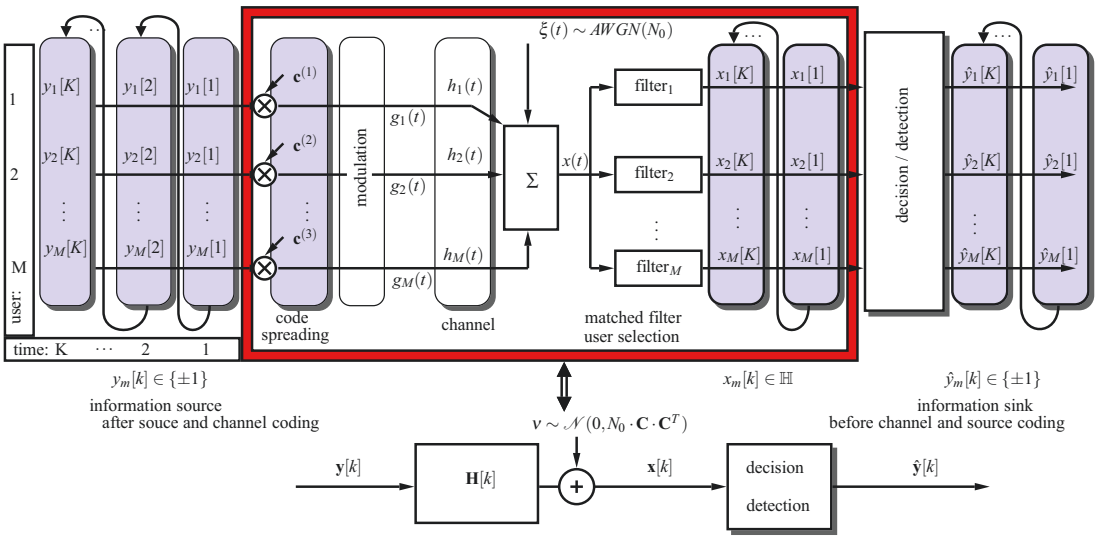


Figure 41: Transceiver system using spread codes for multiple access

denotes the received sequence:

$$\mathbf{x} = \mathbf{H} \cdot \mathbf{y} + \nu \tag{3.20}$$

There are parametric and non parametric approaches to this problem. The parametric approach is based on the knowledge or approximation of the system parameters. If the channel parameters are known, the detection of digital messages in a CDMA system leads to a quadratic optimization [162, 65, 132]. Thus the optimal detection rule can be given as

$$\mathbf{y}_{opt} = \underset{\mathbf{y}\in\{\pm 1\}^L}{\operatorname{argmin}} \mathbf{y}^T\mathbf{W}\mathbf{y} - 2\mathbf{y}^T\mathbf{b} \tag{3.21a}$$

$$\mathbf{W} = \mathbf{H}^T\boldsymbol{\Sigma}^{-1}\mathbf{H} \tag{3.21b}$$

$$\mathbf{b} = \mathbf{H}^T\boldsymbol{\Sigma}^{-1}\mathbf{x} \tag{3.21c}$$

Knowing the codes assigned to the different users of the system and the impulse response of the channel, the detection is clearly a UBQP problem. Thus the optimal detection at the output of such systems is proved to be an NP hard problem [162, 65, 132, 108, 7, 79]. In this case, MLD proves to be of exponential complexity as a function of the number of users and the length of the transmitted sequence [162].

Among the parametric detectors, in order to reduce the detection complexity, SD [96], DF [96], convolutional coding [22], FH type methods [16] and HNN based detectors [79, 80] have been developed, because they can yield a reasonably high performance with limited complexity. For MIMO systems Özdemir and Gürbüz provided a detailed analysis on employing THP MIMO scheme [125]. However, in order to apply these methods one needs the exact channel characteristics. In the lack of this data, one to estimate the unknown channel characteristics based on a training sequence. For indoor environments Kahveci proposed a Max-Log-MAP based technique for channel estimation [88]. For UWB systems Islam, Ameen, and Kwak proposed a Quasi-Newton based iterative algorithm for estimating the channel characteristics [83]. Using

MIMO techniques, classically with SS-MC-MA, each user spreads its data symbols on a specific subset of adjacent or multiplexed subcarriers, to facilitate the channel estimation [16]. However, these methods prove to be rather tedious. That is the reason suggesting user grouping to decrease the computation burden of the ML-MUD for OFDMA-CDMA systems as put forward by Sacchi and Panizza [142]. As a result, in order to implement efficient communication in CDMA, one needs a fast and near-optimal solver of UBQP.

In the following section we analyze the performance of our proposed new algorithms when they are deployed to solve the MUD problem.

Performance analysis for MUD

In this subsection we present the numerical performance of traditional detectors and compare them with the performance of the new algorithms on MUD. In the simulations we used 31 length Gold sequences [56] as spread codes. The channel model for each user was computed based on the COST 207 [35] Typical Urban model with 12 taps. The main parameters for the channel settings were the following:

- the speed of the mobile stations were taken as v 0 01m/s to ensure quasi static channel throughout the simulation;
- the carrier center frequency was set to f_c 900MHz;
- and the bitrate for each user was taken as R 1Mbit/s.

These parameters were chosen to generate a channel with strong frequency selectivity and ISI phenomenon.

The following figure indicates a typical user channel response:

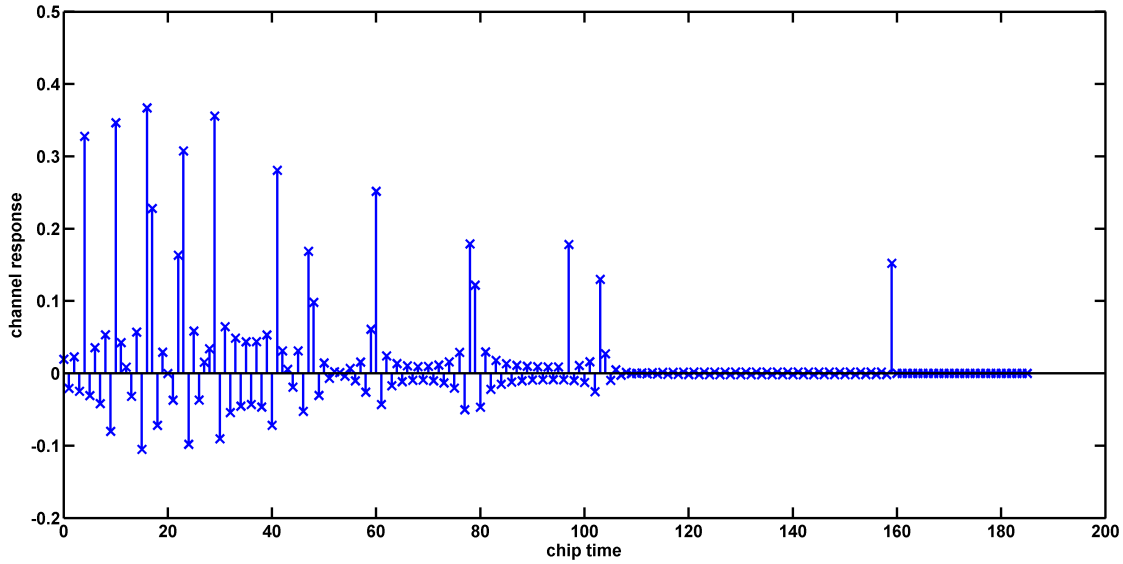


Figure 42: A typical channel response measured for chip time unit

In the simulation we sent information blocks containing 6 symbols for each user, followed by a guard time period [162]. The performance has been analyzed by using the bit error rate on the one hand, and the achieved BQP objective function value, on the other.

DOI:10.15774/PPKE.ITK.2019.006

In order to avoid large values getting out of the range of visualization, in Figure 44 for each SNR, we give subtracted values (i.e. the value of the objective function achieved by a given method minus the smallest value among the outputs of the different algorithms at the same SNR).

In the simulations, the following detectors were used [96]: “threshold” is the plain sign decision rule following the matched filter which filters the symbols with the appropriate code. The “invFilter” is the generalized Zero-Forcing detector trying to equalize the channel and cancel the multi user interference. The “QR” and the “DF” variant of this algorithm uses the QR factorization and reformulation of the problem, and a decision feedback respectively. The “MMSE” is the minimum mean square detector and the sphere detector is denoted with the abbreviation “SD”. In the following figures the performance measures are shown for an unsaturated multi user configuration. We used M 7 users to communicate simultaneously. It can be seen that two of

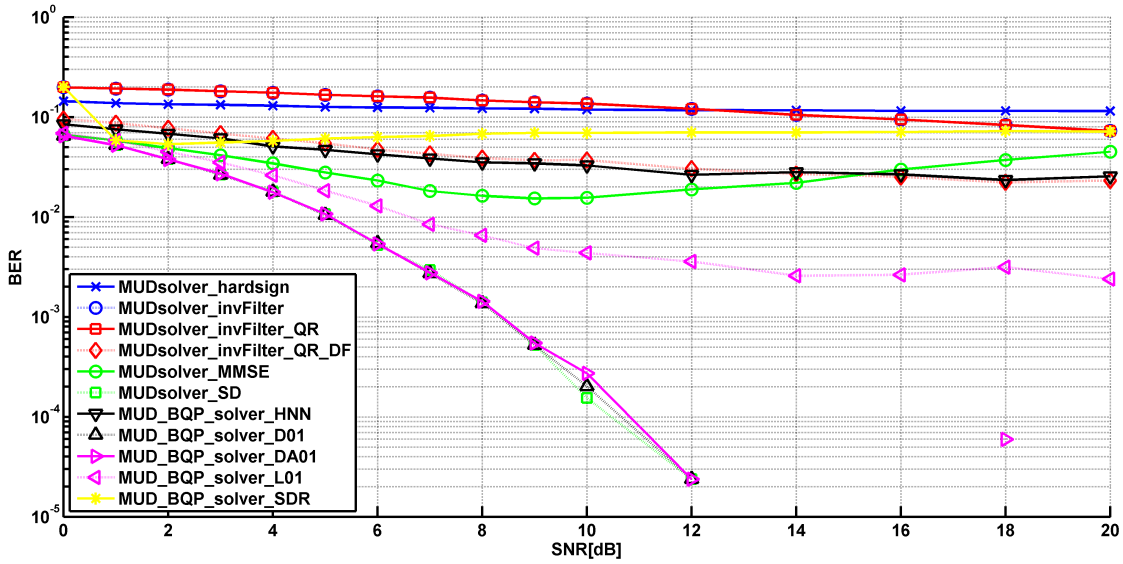


Figure 43: BER performance of the algorithms for 7 users

our solvers perform as well as the Sphere Detector and the other traditional methods perform very poorly under this condition. From the other figure it can be seen that in terms of objective function value for almost all SNR values the “D01” algorithm and the “SD" performs best, but the “DA01” algorithm performs almost identically to them. Referring back to the previous chapter we recall that “DA01" needs much smaller time to reach its candidate solution than the “D01”.

The sphere detector gives a very good quality solution, however it converges much more slowly than the DA01. Furthermore, our proposed algorithms lend themselves to easy parallel implementation.

DOI:10.15774/PPKE.ITK.2019.006

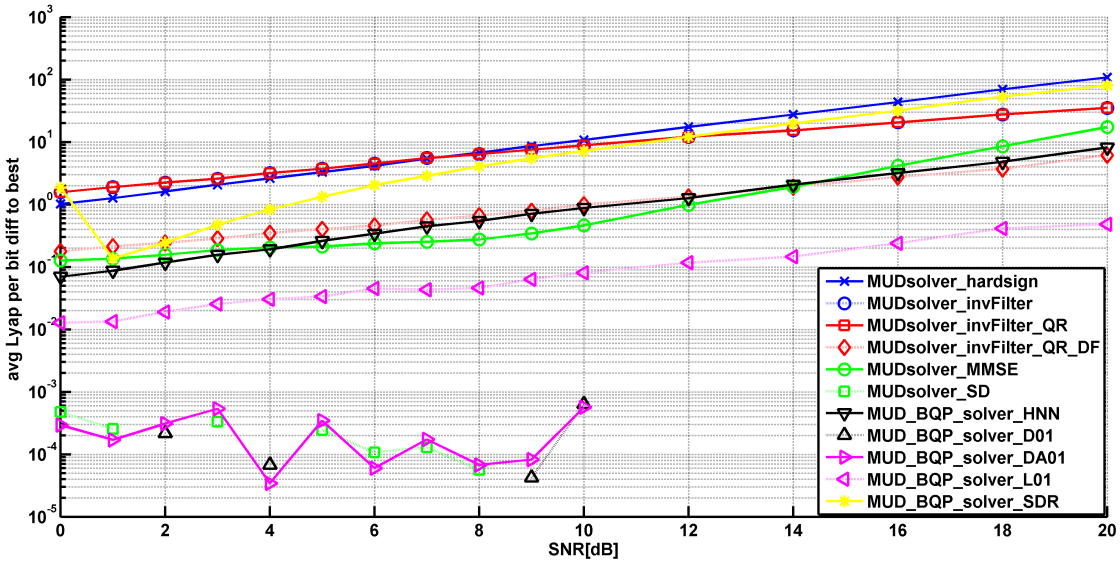


Figure 44: Objective function performance of the algorithms for 7 users

3.7 Conclusion

In this section I have introduced some novel hypergraph based algorithms for solving the UBQP problem which lend themselves to easy parallelism. The basic idea behind the new methods is the partition of the UBQP which results in an aggressive dimension reduction. In this new framework the traditional solvers can reach improved solutions because they work in smaller dimension spaces, furthermore the new approach can give rise to more efficient parallel implementations.

The proposed algorithms operate on two levels:

- Each vertex of the hypergraph represents a fully defined UBQP which can be solved by any traditional UBQP solver.
- Moving from one vertex to another vertex the new method reduce the dimension, as a result the solver can achieve fast solution in a lower dimension search space.

The algorithms have been tested on benchmark problems taken from the ORLIB library and they achieved better performance than traditional solvers of similar complexity. The new methods have also been applied to special current problems of communication such as MUD and scheduling.

In the case of MUD, they achieve similar BER as the traditional methods but the run times needed for the similar quality solutions are shorter.

In the case of scheduling, better TWT can be obtained by the proposed new algorithms. Furthermore the suggested algorithms are more robust regarding the choice of the parameters of the BQP under linear constraints. Due to this robustness we do not have to search for the optimal parameter set because the choice of the parameters will not have a great impact on the solution. Thus solutions can be obtained faster.

Further improvements on the algorithms can be achieved by fine tuning the selection criterion of the vertices. The speed and accuracy can also be improved by using faster solvers. Finally as demonstrated the parallel implementations can have a great impact on the speed proportional to the number of workers used.

DOI:10.15774/PPKE.ITK.2019.006

4 Thesis group III - near Bayesian performance non-parametric detection with Feed Forward Neural Networks

This thesis group elaborates on developing novel encoding techniques for implementing non-parametric, neural network based detectors for pattern recognition on noisy input. This fundamental problem appears in many real world applications like in information extraction in big data context, automated surveillance, speech recognition, content based search or in legacy systems using CDMA to name a few.

I propose an FFNN based algorithm which I present on the MUD problem, but due to the nature of the method it can be easily generalized. In the MUD scenario it is capable of achieving near optimal performance with relatively limited complexity. These new encoding methods on the one hand can increase the processing speed and reduce the complexity of the FFNN based detector, on the other. Furthermore, we demonstrate that an asymptotically optimal detection performance can be achieved by the proposed algorithms. Due to the increased processing rate, the new scheme may further improve SE. Extensive simulations and the corresponding numerical analysis demonstrate that the proposed algorithms yield near optimal performance on real channel models (COST-207). *The corresponding publication of the author is titled "Multi-user detection using non-parametric Bayesian estimation by feed forward neural networks" [159].*

For the non parametric approach to the MUD problem, a FFNN structure can be a good choice because of its general nonlinear approximation capabilities [25, 42, 76, 119, 121, 155] and their inherent parallel architecture. Previous works employing FFNN, used Lagrange optimization procedure to solve MUD [167, 36, 66, 6] using an FFNN either as a full non-parametric detector or in some sub-part of the detection procedure in MIMO or CDMA systems. Also, various structures of NNs, like FFNN, RBFN, RNN were employed as blind detectors for SDMA-OFDM systems and were compared to GA assisted MBER and MMSE detectors [6].

As FFNNs exhibit the property of being universal approximators in ^{p} [63], the optimal MAP decision function can be arbitrarily closely approximated. Furthermore, in this case there is no need for explicit channel knowledge or channel characteristic estimation but the optimal decision function can be learned based on a training sequence. The central issue of deploying FFNN as an optimal MAP detector in modern communication technologies is that the complexity of the network can grow exponentially with respect to the number of different sequences to be detected. As a result, the paper proposes specific encoding techniques in order to minimize the complexity with respect to the neurons at the output layer, thus the processing rate is maximized. This is imperative for improving the SE which is one of the fundamental measures of current wireless technologies.

The results of this section are treated in the following structure:

- In subsection 4.1 we introduce the problem and describe the corresponding model.
- In subsection 4.2 we introduce an FFNN as a non-parametric MAP detector.
- In subsection 4.3 we propose the encoding mechanism which will minimize the FFNN complexity and, on the other hand, maximizes processing rate at the receiver.
- In subsection 4.4 we give a performance analysis based on extensive simulations.

DOI:10.15774/PPKE.ITK.2019.006

- Finally in subsection 4.5 we draw some conclusions.

4.1 Non parametric approach to Multiuser Detection - Optimal decision as a maximum search problem

As it was stated in subsection 3.6 the MUD problem the received signal can be formulated as

$$\mathbf{x} \quad \mathbf{H} \ \mathbf{y} \qquad \text{(3.20 revisited)}$$

In the most general case the optimal decision after receiving a sequence $\mathbf{x}$ is symbol $\hat{\mathbf{y}}$ which is the most probable that had been sent through the channel, as it will minimize the BER [162, 27]. This is also called the Bayesian or MAP decision, which is given as follows:

$$\hat{\mathbf{y}}_{opt} \quad f_{opt} \ \mathbf{x} \quad \underset{\mathbf{y} \quad 1 \ L}{\operatorname{argmax}} \quad Y \quad \mathbf{y} \ X \quad \mathbf{x} \tag{4.1}$$

Note that it can be assumed that every message sequence $\mathbf{y}$ occurs with the same probability. This is reasonable if the system uses a reasonably good source coding mechanism. If a uniform source distribution is assumed, then the MAP decision (4.1) is equivalent to (4.3), the ML decision [27].

$$\text{assuming} \quad Y \quad \mathbf{y} \quad 1 \ 2^L \tag{4.2a}$$

$$\hat{\mathbf{y}}_{opt} \quad f_{opt} \ \mathbf{x} \quad \underset{\mathbf{y} \quad 1 \ L}{\operatorname{argmax}} \quad X \quad \mathbf{x} \ Y \quad \mathbf{y} \quad Y \quad \mathbf{y} \tag{4.2b}$$

$$\hat{\mathbf{y}}_{opt} \quad f_{opt} \ \mathbf{x} \quad \underset{\mathbf{y} \quad 1 \ L}{\operatorname{argmax}} \quad X \quad \mathbf{x} \ Y \quad \mathbf{y} \tag{4.3}$$

Since the conditional probability is a binary quadratic expression given as (4.4) (for further details see [108, 162, 65, 132])

$$X \quad \mathbf{x} \ Y \quad \mathbf{y} \quad 2 \quad ^{\frac{L}{2}} \quad ^{\frac{1}{2}} \exp \quad \frac{1}{2} \ \mathbf{x} \quad \mathbf{H}\mathbf{y}^{\ T} \quad ^{1} \ \mathbf{x} \quad \mathbf{H}\mathbf{y} \tag{4.4}$$

the optimal decision reduces to an UBQP task given as follows (see also (3.21)):

$$\hat{\mathbf{y}}_{opt} \quad f_{opt} \ \mathbf{x} \quad \underset{\mathbf{y} \quad 1 \ L}{\operatorname{argmin}} \ \mathbf{x} \quad \mathbf{H}\mathbf{y}^{\ T} \quad ^{1} \ \mathbf{x} \quad \mathbf{H}\mathbf{y} \tag{4.5}$$

Please note that even if all the parameters of the system are known this decision rule is of exponential complexity with respect to the length of the transmitted sequences, 2^L [7, 96]. Furthermore there are also scenarios where one cannot employ the techniques that approximate the channel parameters, however a non-parametric (blind) detector approach can still be used.

As a result, instead of first identifying the unknown system parameters and then introducing an exponential complexity search algorithm or some sub-optimal methods of polynomial complexity [86, 96] I rather estimate the original conditional probabilities in (4.1) by an FFNN in order to implement the MAP decision.

DOI:10.15774/PPKE.ITK.2019.006

The vector of conditionals at (4.1) is denoted with

$$\mathbf{p}(\mathbf{y}|\mathbf{x}) = \mathbb{P}(Y = \mathbf{y}|X = \mathbf{x}) = \left[p(\mathbf{y}^{(1)}|\mathbf{x}), p(\mathbf{y}^{(2)}|\mathbf{x}), \ldots, p(\mathbf{y}^{(N)}|\mathbf{x})\right]^T, \tag{4.6}$$

where $\mathbf{y}^{(i)} \in \{\pm 1\}^L, i = 1, \ldots, 2^L$ denotes the i^{th} binary sequence. As a result, the MAP decision can be carried out by searching for the maximum among the components of this vector $\mathbf{p}(\mathbf{y}|\mathbf{x})$. Please also note that computing this probability vector directly is also of exponential complexity, since $N = 2^L$.

$$\hat{\mathbf{y}}_{opt} = \mathbf{y}^{(i)}: \quad i = \underset{n \in 1 \ldots N}{\operatorname{argmax}}\, p(\mathbf{y}^{(n)}|\mathbf{x}) \tag{4.7}$$

The block diagram of this exponential complexity optimal detector can be seen in Figure 45

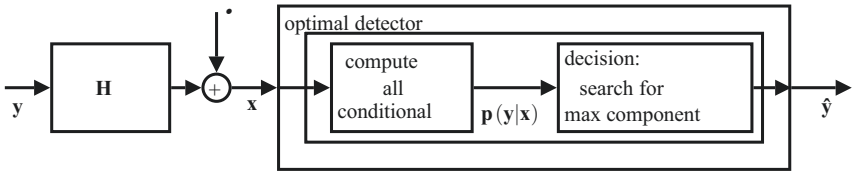


Figure 45: Flow graph representation of the optimal detector

4.2 Application of the FFNN as an optimal detector

In this part I am going to demonstrate that an FFNN can perform optimal non-parametric MAP decision by using a specially encoded training sequence $\mathbf{s}^{(k)}$ [100] and also going to introduce the general concept of encoding via the training set.

The use of FFNN is justified by the fact that it is a universal approximator and thus can represent the conditioned expected value function [25, 76]. Furthermore, since the weights are optimized by learning, FFNN can present an optimal non-parametric estimation which is mandatory in communication technologies where the channel characteristics are unknown [65, 62].

Note that the error function of such network can be described as:

$$\mathbf{w}_{opt}^{(l)}: \min_{\mathbf{w}} \sum_{k=1}^{l} \left\| Net\left(\mathbf{x}^{(k)}, \mathbf{w}\right) - \mathbf{s}^{(k)} \right\|^2. \tag{4.8}$$

and the network asymptotically approximates the conditional expected value:

$$\lim_{l \to \infty} \mathbb{E}\left\| Net\left(\mathbf{x}, \mathbf{w}_{opt}^{(l)}\right) - \mathbb{E}(\mathbf{s}|\mathbf{x}) \right\|^2 = 0 \tag{4.9}$$

which is

$$Net(\mathbf{x}, \mathbf{w}_{opt}) = \mathbb{E}(\mathbf{s}|\mathbf{x}) = \sum_{i=1}^{N} \mathbf{s}^{(i)} p(\mathbf{y}^{(i)}|\mathbf{x}) = \mathbf{S} \cdot \mathbf{p}(\mathbf{y}|\mathbf{x}) \tag{4.10}$$

Let us assume that we assign a specific binary sequence $\mathbf{s}^{(i)} \in \{0,1\}^{N \times 1}$ to every transmitted message $\mathbf{y}^{(i)} \in \{\pm 1\}^{L \times 1}, i = 1 \ldots N$. Please note that the dimension of such vectors are $N = 2^L$.

DOI:10.15774/PPKE.ITK.2019.006

This mapping $\mathbf{y}^{\,i}$ $\mathbf{s}^{\,i}$ now is defined as follows:

$$s_j^{\,i} \quad \begin{array}{ll} 1 & \text{if } i \quad j \\ 0 & \text{if } i \quad j \end{array} \tag{4.11}$$

Since (4.11) describes a code matrix which is in the present case the identity matrix, the expectation becomes

$$\mathbf{s}\,\mathbf{x} \quad \mathbf{I}\,\mathbf{p}\,\mathbf{y}\,\mathbf{x} \quad \mathbf{p}\,\mathbf{y}\,\mathbf{x} \tag{4.12}$$

Since the FFNN approximates the conditional expected function $\mathbf{s}\,\mathbf{x}$ $\mathbf{S}\,\mathbf{p}\,\mathbf{y}\,\mathbf{x}$ $\mathbf{I}\,\mathbf{p}\,\mathbf{y}\,\mathbf{x}$, the output is $\mathbf{p}\ \mathbf{y}\ \mathbf{x}$. As a result, our task is just to search for the maximum among these probabilities.

$$\hat{\mathbf{y}}_{opt} : \underset{i\ \ 1\ \ N}{\operatorname{argmax}}\,\mathbf{p}_i\ \mathbf{y}\ \mathbf{x} \quad \underset{i\ \ 1\ \ N}{\operatorname{argmax}}\,p\ \mathbf{y}^{\,i}\ \mathbf{x} \tag{4.13}$$

The block diagram of the optimal FFNN detector is depicted by Figure 46. Please, note that the

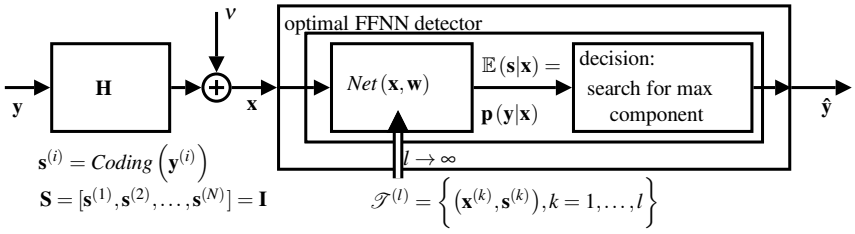


Figure 46: Flow graph representation of the optimal detector

scheme described above is truly non-parametric, i.e. only a training sequence is needed and the optimal decision can be carried out without any a-priori knowledge about system parameters (e.g. channel characteristics etc.). However, this scheme requires a very large FFNN as the vectors $\mathbf{s}$, which are the output vectors of the FFNN, are of dimension 2^L.

Therefore, we must modify the encoding mechanism (the mapping of the transmitted binary sequences $\mathbf{y}$-s into shorter code vectors $\mathbf{s}$-s) to obtain smaller size of neural network which in turn will also increase the processing rate. Therefore, the challenge is to find the optimal code set $\mathbf{S}$ which will keep complexity under a reasonable bound. One must note, though, that in general coding scheme the maximum conditional probability may not be directly read out from the outputs. Thus, we set on developing such coding set which will still enable the use of MAP (i.e. the maximum probability can be uniquely identified) but the complexity gets much lower than the one using code words of length 2^L.

4.3 New coding techniques for minimum complexity FFNN

Let us choose an arbitrary bijective mapping $\mathbf{y}^{\,i}$ $\mathbf{s}^{\,i}$ which maps the sent symbols into the training set targets, called code words and let us arrange the code words into a matrix as follows:

$$\mathbf{S} \quad \mathbf{s}^{\,1}\ \mathbf{s}^{\,2} \quad \mathbf{s}^{\,N} \quad C\ \ N \tag{4.14}$$

DOI:10.15774/PPKE.ITK.2019.006

where the code words are interpreted as column vectors. Similarly to (4.6), let us denote the conditional probability vector of the code words as:

$$\mathbf{p}_{\mathbf{s}|\mathbf{x}} = \left[p(\mathbf{s}^{1}|\mathbf{x}) \;\; p(\mathbf{s}^{2}|\mathbf{x}) \;\; \ldots \;\; p(\mathbf{s}^{N}|\mathbf{x}) \right]^{T} \in \{0,1\}^{N \times 1} \tag{4.15}$$

Due to the bijective nature of the mapping, the following is true: $\mathbf{y}^{i} = \mathbf{x} \Leftrightarrow \mathbf{s}^{i} = \mathbf{x}$. Thus, the expectation of (4.9) can be rewritten as

$$\mathbf{s}(\mathbf{x}) = \sum_{i=1}^{N} \mathbf{s}^{i} p(\mathbf{s}^{i}|\mathbf{x}) = \sum_{i=1}^{N} \mathbf{s}^{i} p(\mathbf{y}^{i}|\mathbf{x}) = \mathbf{S}\,\mathbf{p}_{\mathbf{y}|\mathbf{x}} \tag{4.16}$$

After training $\mathbf{s}(\mathbf{x})$ will appear at the output of FFNN. This general coding capability of the FFNN via the training set is depicted in Figure 47. The block diagram of the detector using

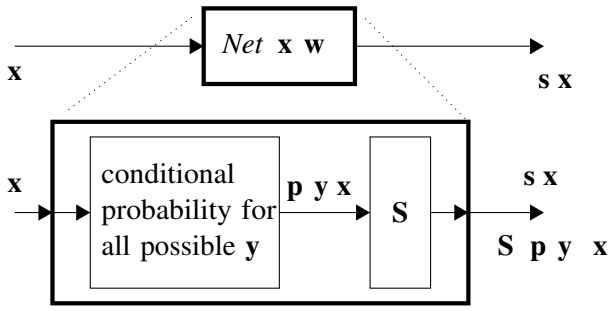


Figure 47: Equivalence of the FFNN with an encoding

an arbitrary code matrix $\mathbf{S}$ is depicted in Figure 48. Our objective is to develop such codes

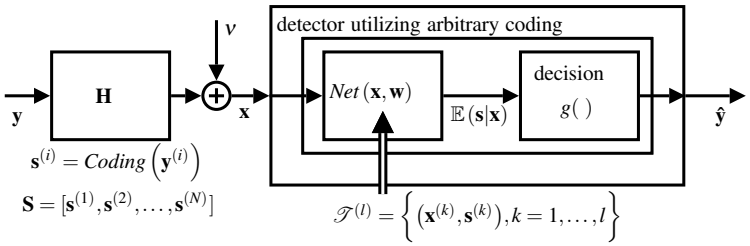


Figure 48: Flow graph representation of the detector using an arbitrary encoding

$\mathbf{s}^{i}$ $i = 1 \ldots N$ which under some conditions will reproduce the MAP decision. This objective must be achieved effectively under the following constraints:

- The length of the code words are small $L \ll N$, because then the number of output neurons are small.
- The decision procedure $\hat{\mathbf{y}} = g(\mathbf{s}(\mathbf{x}))$ is of small complexity and can be easily parallelized.

If we reduce the dimension too aggressively, then we cannot reproduce the MAP decision.

Coding by interval splitting with parameter 2

With this coding the number of outputs in the FFNN can be reduced to L instead of $N = 2^{L}$, furthermore, the processing rate is increased accordingly. The objective of the method is to develop a code $\mathbf{s}^{i}$ $i = 1 \ldots N$, which yields a conditioned expected value vector $\mathbf{s}(\mathbf{x})$ in such

DOI:10.15774/PPKE.ITK.2019.006

a way that the maximum conditioned probability can be obtained by logarithmic search. In order to obtain the unique MAP decision we have to assume the following properties of the conditional probabilities. Let us define a set of indices $E \quad 1 \quad N \quad E \quad N \; 2$ which splits all the indices into to disjoint halves. We assume that for our chosen index set E the following holds:

$$\begin{aligned} & j \quad \max_{i \; 1 \; N} p\left(\mathbf{s}^{i} \; \mathbf{x}\right) \\ & \text{if } j \quad E \\ & \quad p\left(\mathbf{s}^{k} \; \mathbf{x}\right) \qquad\qquad p\left(\mathbf{s}^{i} \; \mathbf{x}\right) \\ & k \; E \qquad\qquad i \quad 1 \; N \quad E \end{aligned} \tag{4.17}$$

This means that the maximum can be sought by logarithmic search if we investigate L possible disjoint half index set pairings by summing them up and comparing them against each other pair wise. For example this interval halving can be obtained by first summing the first half of $\mathbf{p} \; \mathbf{y} \; \mathbf{x}$ with $\quad 1$ and the other half with $\quad 1$ weights. Then we split the previously obtained halves and sum the first and second half with $\quad 1$ and $\quad 1$ weights, respectively. Furthermore, we repeat this interval halving mechanism until the splitting zooms down to an individual component, thus locating the index of the symbol with maximum probability.

The code matrix can be expressed in a closed form as follows:

$$S_{i \; j} \quad s_{i}^{\; j} \quad \text{sgn} \quad \sin \quad 2 \quad 2^{\; i \; 1} \quad \frac{j}{N \quad 1} \qquad i \quad 1 \quad L \;\; j \quad 1 \quad N \tag{4.18}$$

Based on this coding scheme the decision function reduces to the traditional sgn function, since for each stage we only need to decide if the sum of the components encoded with $\quad 1$ or the sum of the components encoded with $\quad 1$ is bigger in absolute value. Thus the sign of that particular stage will indicate in which half is the component having the maximum probability. In this way $L \quad C \quad \log_2 N \qquad N$ as a result, this coding reduces the number of the output of the FFNN to L and also increases the processing rate. So instead of a linear search the decision algorithm is

$$\hat{\mathbf{y}} \quad \text{sgn} \quad \mathbf{s} \; \mathbf{x} \tag{4.19}$$

The only condition that must be fulfilled in order for this encoding scheme to perform as well as the optimal MAP decision is (4.17) and the numerical simulation will demonstrate that this condition is satisfied with a relatively high probability.

Examples for coding by interval splitting with parameter 2 To give an example we present this encoding with parameters $L \quad C \quad 3, N \quad 2^3 \quad 8$:

$$\mathbf{S} \quad \mathbf{s}^{1} \; \mathbf{s}^{2} \qquad \mathbf{s}^{N} \qquad \begin{matrix} -1 & -1 & -1 & -1 & +1 & +1 & +1 & +1 \\ -1 & -1 & +1 & +1 & -1 & -1 & +1 & +1 \\ -1 & +1 & -1 & +1 & -1 & +1 & -1 & +1 \end{matrix} \tag{4.20}$$

The first row represents the step where we sum up the two halves with $\quad 1$ weights, the second row represents the step when we sum up interval fourths with appropriate weights and the last

DOI:10.15774/PPKE.ITK.2019.006

row represents the last step in this example where we sum up individual interval eights with the appropriate 1 weights. For example in (4.21) the case is detailed if we assume that $p\ \mathbf{y}^{4}\ \mathbf{x}$ is the component with the maximum probability. In this case the first row lets us know that the maximum component has an index from 1 2 3 4 , the second row tells that the maximum component has an index from 3 4 7 8 , the third row tells that it has an index from 2 4 6 8 . Combining these yields to index 4.

$$\hat{\mathbf{y}} \quad \text{sgn} \quad \mathbf{s}\,\mathbf{x} \qquad \begin{matrix} -1 \\ +1 \\ +1 \end{matrix} \tag{4.21a}$$

$$\mathbf{S}\ \mathbf{p}\ \mathbf{y}\ \mathbf{x} \qquad \begin{matrix} -1 & -1 & -1 & -1 & +1 & +1 & +1 & +1 \\ -1 & -1 & +1 & +1 & -1 & -1 & +1 & +1 \\ -1 & +1 & -1 & +1 & -1 & +1 & -1 & +1 \end{matrix} \qquad \begin{matrix} p\ \mathbf{y}^{1}\ \mathbf{x} \\ \\ p\ \mathbf{y}^{4}\ \mathbf{x} \\ \\ p\ \mathbf{y}^{8}\ \mathbf{x} \end{matrix} \tag{4.21b}$$

Handling the error if the assumption does not hold. However there are received $\mathbf{x}$ points for which assumption (4.17) does not hold. In this case our detection mechanism yields to an erroneous symbol (in a sense that it is not the most likely). To show this phenomenon here is an example where L 2. In Figure 49 one can see the four received symbols if no noise is present: $\mathbf{H}\mathbf{y}^{i}$ i 1 4. These are marked with small squares. Around them contour curves of the additive noise are depicted. There are four distinct areas where assumption (4.17) does not hold. These are marked with shaded areas colored to blueish red, blueish black, greenish red and greenish black. All areas are colored to the main color for which the decision is made and to the secondary color for which the decision should have been made. E.g. a received symbol in the blueish red area will be decoded as a “blue" sent symbol instead of a “red". For the sake of the example I have chosen a received symbol $\mathbf{x}$ 0 435 0 685 ^{T} in the blueish red area. In the box at the right side of the symbol we denoted the corresponding conditional probabilities:

$$\mathbf{p}\ \mathbf{y}\ \mathbf{x} \qquad \begin{matrix} p\ \mathbf{y}^{1}\ \mathbf{x} & p\ \mathbf{y}^{2}\ \mathbf{x} & p\ \mathbf{y}^{3}\ \mathbf{x} & p\ \mathbf{y}^{4}\ \mathbf{x} \end{matrix}^{T}$$

$$0\ 4406 \quad 0\ 0002 \quad 0\ 3364 \quad 0\ 2229^{T}$$

One can see that (4.17) does not hold, since $p\ \mathbf{y}^{1}\ \mathbf{x}$ $p\ \mathbf{y}^{2}\ \mathbf{x}$ $p\ \mathbf{y}^{3}\ \mathbf{x}$ $p\ \mathbf{y}^{4}\ \mathbf{x}$, but $\mathbf{y}^{1}$ has the maximum probability. In this case

$$\mathbf{s}\,\mathbf{x} \qquad \mathbf{S}\ \mathbf{p}\ \mathbf{y}\ \mathbf{x} \qquad \begin{matrix} -1 & -1 & +1 & +1 \\ -1 & +1 & -1 & +1 \end{matrix} \qquad \mathbf{p}\ \mathbf{y}\ \mathbf{x} \qquad \begin{matrix} 0.1185 \\ -0.5539 \end{matrix}$$

Thus our detection algorithm, sgn $\mathbf{s}\,\mathbf{x}$ will choose $\hat{\mathbf{y}}$ 1 1 ^{T} instead of $\hat{\mathbf{y}}_{opt}$ $\mathbf{y}^{1}$ 1 1 ^{T}. Note that by introducing assumption (4.17), we also introduce an inherent error compared to the theoretical optimum even if our network perfectly learns the expectation.

DOI:10.15774/PPKE.ITK.2019.006

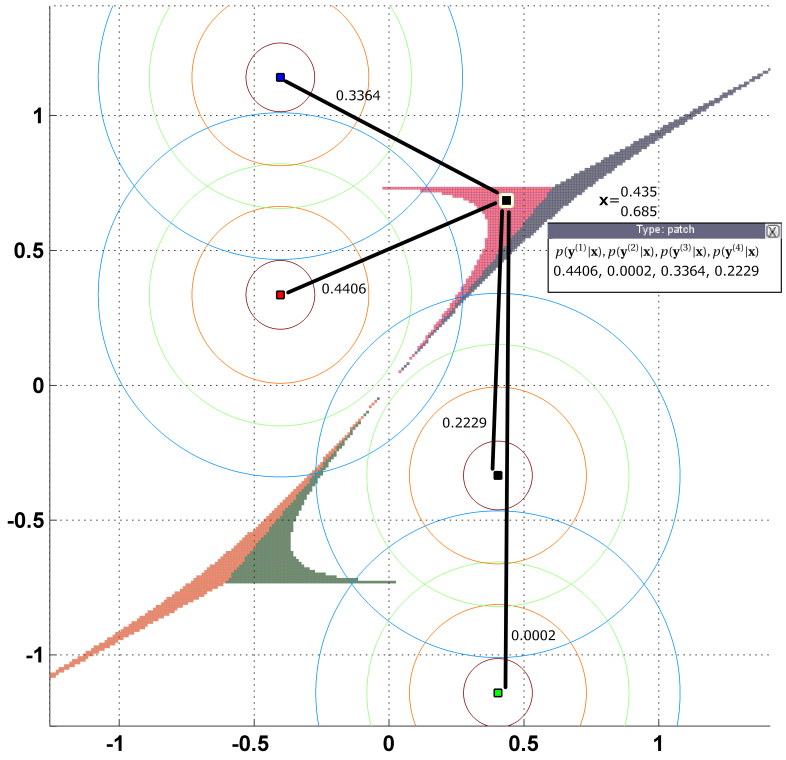


Figure 49: Coding error of the interval halving method

Similarly, to the previous transformation it is possible to generalize the proposed coding mechanism by using higher order logarithmic search. This can be obtained by splitting the index set into smaller partitions (e.g. thirds, fourths, etc.). The objective of this generalized method is to give a scalable balance between the computational complexity and the achievable theoretical performance. This generalized version of the coding, enabling logarithmic search, strikes a good compromise between $dim\ \mathbf{s}$ and performance. On the one hand if we utilize a higher order logarithmic search then constraint (4.17) can be weakened, thus the performance will fall closer to the MAP decision, but at the same time it increases the complexity of the FFNN.

THESIS III.1 (blind detection by interval halving and FFNN)**.** *I have defined an FFNN based blind detector for the MUD problem, which lends itself to easy parallelization and can perform optimally under the constraint defined in* (4.17)*. In* (4.18)*, I give the linear encoding based on interval halving which is used to generate a training set for an FFNN and in* (4.19) *I give the low complexity decision function which is to be employed on the output of the net. I have shown that the detector performs near optimally on the investigated MUD scenarios described in subsection 4.4.*

The thesis is restated in a self consistent way in Appendix A at Thesis III.1 (page 88).

4.4 Numerical performance analysis

In this chapter we investigate the performance of the proposed methods compared to other well known detectors. We also present the network architecture and training parameters that were used for the simulation. Our detailed performance analysis is based on plotting the BER

versus the SNR. This is the fundamental measure which captures the quality of service in wireless communication systems. Furthermore, to compare our methods with some traditional detection techniques we also plotted the BER versus SNR curves for the following three basic linear algebra based detection algorithms [96], for a SD and for an UBQP based heuristic.

- Threshold detector: sgn $\mathbf{x}$ sgn $\mathbf{Hy}$
- ZF detector: sgn $\mathbf{H}^{-1}\mathbf{x}$
- MMSE detector: sgn $\mathbf{H}\overline{N_0}\mathbf{I}^{-1}\mathbf{x}$
- SD: A common version of the algorithm was implemented which is a well-known ML type detector [86, 96, 27].
- "DA01": is an UBQP based heuristic which builds up a solution adding a dimension iteratively. [160]

The parameters of the communication system under investigation are given as follows:

- We used 31 length Gold sequences [56] as spread codes.
- The channel model for each user was computed based on the COST 207 models [35]. Four models were used, namely "COST207 Hilly Terrain 6 tap alternative", "COST207 Rural Area 6 tap", "COST207 Typical Urban 12 tap" and "COST207 Bad Urban 12 tap" models.
- The main parameters for the channel settings are as follows: (i) the speed of the mobile stations were taken as v 0 01m/s to ensure quasi static channel throughout the simulation; (ii) the carrier center frequency was set to f_c 900MHz, because these models were calibrated to use that band; (iii) and the bitrate for each user was taken as R 1Mbit/s.

We chose these specific reference channel models with these parameters, because they cover the range from a mildly distorted to a channel which has very strong frequency selective fading, which yields to strong ISI.

4.4.1 Network architecture and training parameters

Our network architecture is a 3 hidden layer structure depicted by Figure 50, because these networks with sufficiently large number of neurons in their hidden layers can approximate any continuous function [25, 42, 119], such us our target function $\mathbf{s}\ \mathbf{x}$. We choose the size of the

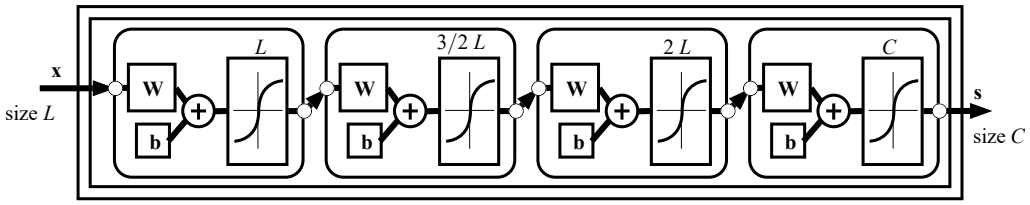


Figure 50: The architecture of the FFNN used in simulations

hidden layers proportional to the dimension of the problem. In the first layer we used L neurons, in the second layer 3 2 L and in the third hidden layer 2 L neurons. At the output layer there are C neurons which equals to the size of the target codewords.

In all layers we used hyperbolic tangent sigmoid transfer functions. For training method we have adopted a scaled conjugate gradient backpropagation type algorithm with scaled inputs

DOI:10.15774/PPKE.ITK.2019.006

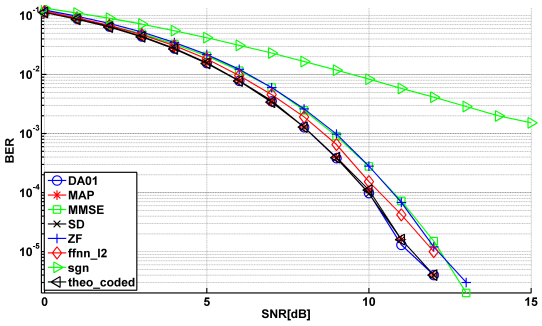
and outputs. The performance function of the backpropagation algorithm was the mean square error between the outputs of the network and the target values over the elements of the training set. There were no validation set used, since the goal of the training is not to fully reproduce the targets, but to approximate the conditional $\mathbf{s}\ \mathbf{x}$. We ran the training algorithm until the performance gradient was relatively small. We defined our training set to contain W times each transmittable symbol to cover the whole space, where W was typically 50. This number was chosen empirically because this stroke a fair balance between the performance and the training time of the network. We sent these symbols ($\mathbf{y}$) through the channel and stored the received signal ($\mathbf{x}$) as inputs and also mapped the sent symbols into codewords ($\mathbf{s}$) and stored them as the targets as follows:

$$^{W\ N} \qquad \mathbf{x}\ \mathbf{s} \quad \mathbf{x} \quad \mathbf{H}\ \mathbf{y} \qquad \mathbf{s} \quad \textit{Coding}\ \mathbf{y}$$

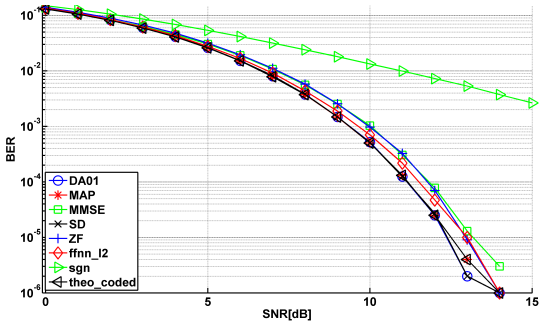
4.4.2 Performance of the new detector

On the one hand in the simulations we computed the exact MAP decision. On the other we calculated the achievable BER by employing the newly developed coding scheme. These are labeled by MAP and “theo coded”. Our new detection algorithm was labeled as “ffnn I2”. The same model was applied as in Figure 41, a CDMA system with M users transmitting K length blocks at once ($L \quad K \quad M$).

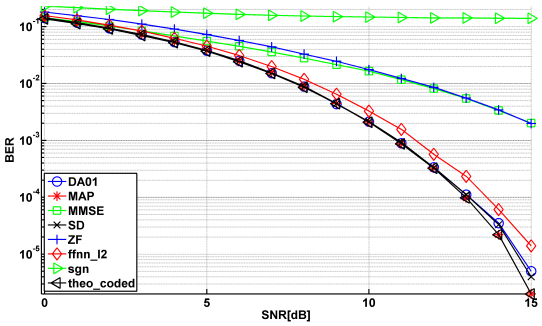
In Figure 51a - Figure 51d the performances are depicted when $M \quad 5$, $K \quad 2$ $L \quad 10$ for different channels. One can see, when the channel changes from the one measured on hilly terrain



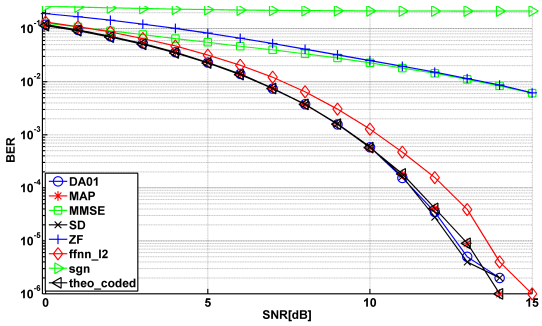
(a) BER using channel COST 207 Hilly Terrain 6 tap alternative



(b) BER using channel COST 207 Rural Area 6 tap



(c) BER using channel COST 207 Typical Urban 12 tap



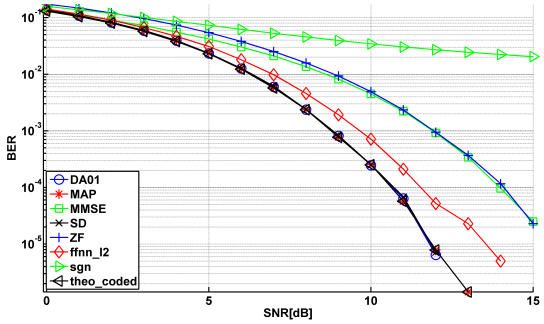
(d) BER using channel COST 207 Bad Urban 12 tap

Figure 51: Performance curves with parameter $L \quad 10$ for four typical channels

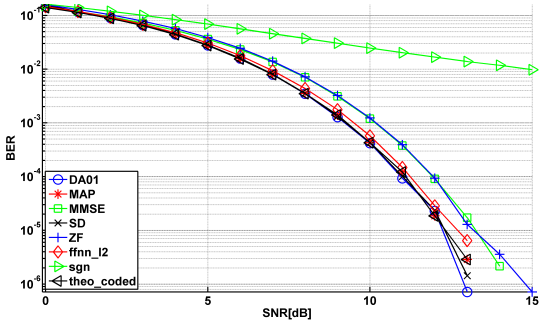
DOI:10.15774/PPKE.ITK.2019.006

to another one measured in urban area, practically can achieve the optimal performance. The difference between the theoretical performance and the actual one is due to the asymptotic nature of learning. Namely the FFNN failed to capture the conditional expected value perfectly. However, it achieves similar BER than the best performing SD. Note that the SD is a parametric detection which needs the channel characteristics as opposed to our proposed method which does not.

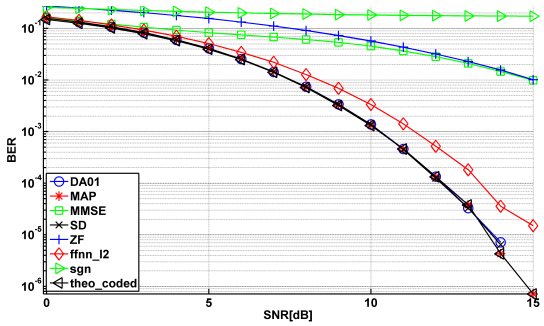
Similarly Figure 52b - Figure 52d show the performance of the same methods for M 7,K 2 and L 14. One can see that "FFNN I2" again produces only slightly worse performance than



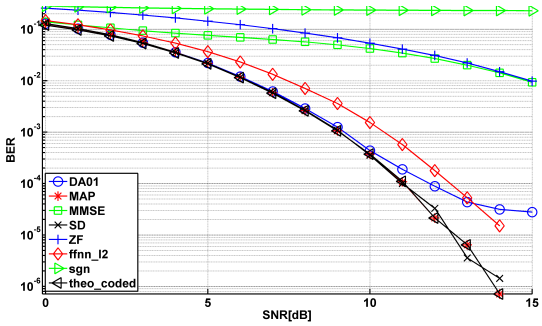
(a) BER using channel COST 207 Hilly Terrain 6 tap alternative



(b) BER using channel COST 207 Rural Area 6 tap



(c) BER using channel COST 207 Typical Urban 12 tap



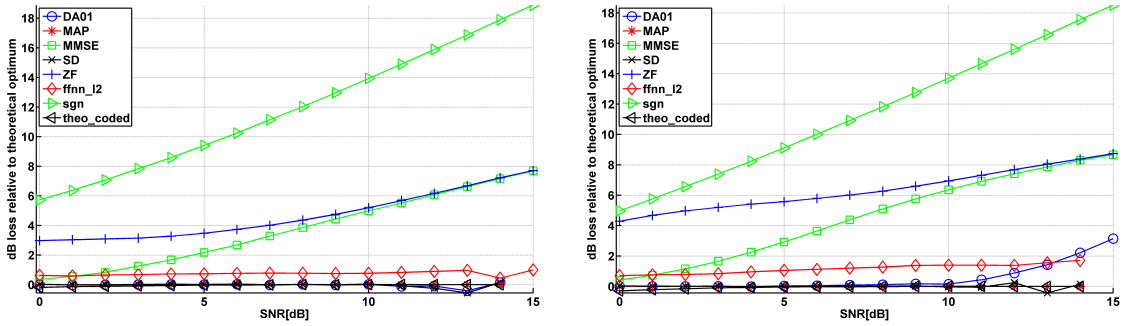
(d) BER using channel COST 207 Bad Urban 12 tap

Figure 52: Performance curves with parameter L 14 for the four channel models

the MAP decision. Again the performance degradation is due to the approximative nature of FFNN and also due to the violation of assumption (4.17). But even in this case the novel method provides low BER with respect to SNR.

We introduce the measure "SNR loss" indicating how much more signal energy is needed by a given method to the same BER as achieved by the MAP. This measure is depicted by Figure 53a and Figure 53b respectively. On average our proposed method performs as well as the MAP decision at a 0.5-1.5 dB lower SNR level. Furthermore, one can see that our new method has a nearly constant dB loss curve in contrast to the ZF and MMSE equalizers. The following Table 5 and Table 6 summarize the performance of the proposed algorithm. In the tables the minimum and maximum SNR loss are indicated compared to the MAP decision for the four channel models. The small negative values in the table appear due to numerical imprecision of the simulation at high SNR levels.

DOI:10.15774/PPKE.ITK.2019.006



(a) SNR loss curves with parameter L 10 for channel COST 207 Bad Urban 12 tap channel

(b) SNR loss curves with parameter L 14 for channel COST 207 Bad Urban 12 tap channel

Figure 53: SNR loss curves for the Bad Urban channel model

Table 5 SNR loss for "Hilly Terrain" and "Rural Area" channels

	"Hilly Terrain"				"Rural Area"			
	L=10		L=14		L=10		L=14	
	min	max	min	max	min	max	min	max
ZF	0 2916	0 9124	1 3035	3 5167	0 0	0 6658	0 4905	1 8582
MMSE	0 0270	0 9675	0 0034	3 5689	0 0268	0 7772	0 0052	0 9608
DA01	0 1445	0 0381	0 1226	0 0581	0 5995	0 0301	0 1501	0 0851
SD	0 0767	0 0615	0 0475	0 0955	0 6	0 2064	0 1072	0 4281
FFNN I2	0 0838	0 6020	0 2552	1 6643	0 0	0 4896	0 0860	0 2695

Table 6 SNR loss for "Typical Urban" and "Bad Urban" channels

	"Typical Urban"				"Bad Urban"			
	L=10		L=14		L=10		L=14	
	min	max	min	max	min	max	min	max
ZF	1 4406	4 9257	3 9235	7 4737	2 9822	7 7086	4 2888	8 7401
MMSE	0 1024	4 9496	0 1010	7 4360	0 3506	7 6913	0 4142	8 6666
DA01	0 0348	0 2744	0 0776	0 1813	0 4131	0 1468	0 0100	3 1431
SD	0 0172	0 2360	0 0776	0 1388	0 5551	0 1468	0 4276	0 2429
FFNN I2	0 4713	0 7130	0 3906	1 5429	0 4449	1 0	0 7063	1 7197

DOI:10.15774/PPKE.ITK.2019.006

4.5 Conclusions

In this section I have proposed a novel encoding scheme for FFNN based detectors in multi-user communication systems. FFNNs were used in order to estimate the conditioned expected value in a non-parametric manner. This approach is justified by the fact that it does not require any a-priory knowledge about the channel characteristics. Furthermore due to the wide scale representation capabilities of FFNNs they can capture the non-linear characteristics of the conditional expected value. The specific coding scheme helps us to identify the maximum conditioned probability (MAP decision) from the estimated conditional expected value. In this case we obtain a generic non-parametric MAP decision which only uses a training set. The estimation of the conditional expected value is obtained via learning.

The advantage of the proposed method is that it can achieve optimum detection performance by carrying out the MAP decision (see Figures 51 - 53) even in the lack of channel parameters. The disadvantage lies with the relatively slow training process, which may slow down the convergence to MAP. Furthermore this may not enable the application of the method to channels exhibiting fast time varying characteristics as the convergence of learning may take longer time than keeping track of the time varying channel. However in the case of stationary characteristics the method yields a very good performance and as a result the system will suffer from only a marginal "SNR loss". Another disadvantage is that without any assumption on the conditional probabilities the method needs exponential complexity. Nevertheless if the conditional probabilities fulfill some mild conditions and with an appropriately chosen coding scheme, this complexity can be significantly reduced as well as the processing rate can be increased.

In this case the proposed detector scheme is of a small complexity architecture which can be easily parallelized and uses a very simple decision function. Furthermore we have also demonstrated that with the new coding schemes almost optimal performance can be achieved with regard to BER vs the SNR.

DOI:10.15774/PPKE.ITK.2019.006

5 Summary of the dissertation and closing remarks

In this dissertation I have given the following answers to the posed questions:

One can efficiently find an appropriate path or tree in a packet switched network which provides a QoS (either bottleneck or additive type). This can be achieved by exploiting the statistical properties of the traffic and transforming the models in such a way, that they become a natural fit for the traditional route finding algorithms or neural networks. Furthermore the precision with which the QoS is met can be scaled at the expense of some bandwidth by applying information theoretic measures.

One can solve efficiently and near optimally the UBQP problem which is present in relevant ICT applications: the scheduling tasks in communication networks (IoT or cloud computing environments), load balancing and MUD for wireless technologies. This problem can be treated in a parallel fashion with the aid of both Feed Forward and Recurrent type neural networks. To achieve this I have demonstrated how to reformulate the problems to fit these algorithms. For the Recurrent type neural network I posed these problems as an "energy based" optimization problem. For the Feed Forward neural networks, I exploited their general approximation capabilities.

One can solve efficiently and near optimally a general pattern recognition problem with the aid of Feed Forward neural networks and a linear encoding technique. I have demonstrated the efficiency of the algorithm on the MUD problem, but it is applicable on a wide range of problems including automated surveillance applications, content based search, speech recognition.

The numerical examples presented, back up my conjecture that these algorithms are in deed applicable and perform efficiently.

Although a lot of aspects were not addressed, this work gives sufficient details, such that it can be used as a basis for further investigation. For example how a physical implementation of such neural networks could speed up finding sub-optimal solutions for these problems. Furthermore it gives a common numerical reference for comparison to other types of algorithms. A possible natural extension of the proposed methods would be to change the currently used neural networks with "deep-learning" based variants, compare the performance and investigate the gains and losses. Certainly if a particular application is to be considered, these algorithms need tailoring. Also note that the appropriate physical architecture may not exist at the time of writing but this work might point to such possible directions.

DOI:10.15774/PPKE.ITK.2019.006

Appendices

A New scientific results and theses of the dissertation

This chapter summarizes - without any proofs - the new scientific results and theses of the dissertation in a self consistent way.

Thesis group I - routing with incomplete information in unicast and multicast scenarios

THESIS I.1 (unicast routing with incomplete information by Gaussian approximation)**.** *I gave a mapping for the link descriptors under the condition that the link descriptors have normal distributions with parameters m and* $\tilde{}_{u\,v}$ $\overline{m}_{u\,v}$ *and also the LAS follows m* $\frac{t_i\ 1\ t_i}{2}$ *in Theorem 1 (restating):*

Theorem 1. *If* $_{u\,v}$ *is a subject to a normal distribution with parameters* $\tilde{}_{u\,v}$ $\overline{m}_{u\,v}$*, then the solution of ARII*

$$\tilde{R} \quad \underset{R \quad s \quad d}{\operatorname{argmax}} \quad _{u\,v \quad R}^{\quad u\,v} \quad \mathrm{T} \qquad \text{(2.3 revisited)}$$

is equivalent to minimizing the objective function

$$\tilde{R} \quad \underset{R}{\operatorname{argmin}} \quad \underset{u\,v \quad R}{m_{u\,v}} \qquad (2.13)$$

by using the Bellman-Ford algorithm in polynomial time.

Using these assumptions the ARII problem can be reduced to a deterministic traditional SPR.

THESIS I.2 (unicast routing with incomplete information by recursive path finder algorithm)**.** *I gave procedures that can find routes in a packet switched network which satisfy the required QoS parameter with a given probability in Algorithm 1 and Algorithm 2 (restating below).*

The algorithms are based on a transformation of the random link descriptors using the large deviation theory which is described in Theorem 2 (restating):

Theorem 2. *Using the logarithm of the moment generating function (log-moment generating function)*

$$_{u\,v} \quad s \quad \ln \quad \exp \quad s \quad _{u\,v} \qquad \ln \quad \exp \quad sx \quad dF_{u\,v} \quad x \qquad (2.20)$$

or in case of a discrete random variable

$$_{u\,v} \quad s \quad \ln \quad \exp \quad s \quad _{u\,v} \qquad \ln \quad \underset{i \quad 1}{\exp} \quad sx_i \quad p_i \qquad (2.21)$$

the solution of the ARII is equivalent with minimizing the objective function

$$\tilde{R} \quad \underset{R}{\operatorname{argmin}} \quad _{u\,v \quad R}^{\quad u\,v} \quad \hat{s} \qquad (2.22)$$

DOI:10.15774/PPKE.ITK.2019.006

where the optimal $\hat{s}$ parameter is

$$\hat{s} \quad \inf_{s} \quad _{u\,v} \quad s \quad sT \tag{2.23}$$

Algorithm 1 Exhaustive-s algorithm

Input: $G\ V\ E\ \ _{u\,v}\ F_{u\,v}\ x\ \ src\ dst$

Define a grid on the set of possible values of s denoted by $s_i\ s_i\ 0\ i\ 1\ M$.

for all $i\ 1\ M$ **do**

Pick s_i .

Perform path selection R_i by an SPR algorithm with link measures $_{u\,v}\ s_i : \ln \exp s_i\ _{u\,v}$.

Based on the selected path R_i determine

$$\hat{s}_i \quad \text{Solve} \quad _{u\,v\ \ \tilde{R}_i} \frac{d\ _{u\,v}\ s}{ds} \quad T\ s \tag{2.28}$$

and calculate the bound

$$B_i : \quad \exp \quad _{u\,v\ \ R_i} \quad _{u\,v}\ \hat{s}_i \quad \hat{s}_i T \tag{2.29}$$

end for

Find the path which belongs to minimal bound

$$\tilde{R}_j : j \quad \operatorname*{argmin}_i B_i \tag{2.30}$$

Output: $\tilde{R}_j$ chosen path between *src* and *dst*

Algorithm 2 The Recursive Path Finder - s Finder Algorithm

Input: $G\ V\ E\ \ _{u\,v}\ F_{u\,v}\ x\ \ src\ dst$

Pick s a positive starting value

compute the path independent s

repeat

Associate measure $_{u\,v}\ s$ to each link $u\ v\ E$.

Perform the SPR algorithm to find the optimal path $\tilde{R}\ s$ for parameter s.

For the obtained $\tilde{R}$ determine $\tilde{s}$ by expression

$$\tilde{s} \quad \frac{^{1}\ T \quad _{u\,v\ \ R}\ a_{u\,v}}{R} \tag{2.40 revisited}$$

$s \quad \tilde{s}$.

until $\tilde{R}\ \tilde{s} \quad \tilde{R}\ s$

Output: $\tilde{R}\ s$ chosen path between *src* and *dst*

DOI:10.15774/PPKE.ITK.2019.006

THESIS I.3 (multicast routing with incomplete information with HNN)**.** *I defined algorithm to find a sub-optimal solution to the multicast routing problem with random link descriptors in Algorithm 3 (restating):*

Algorithm 3 Find optimal tree for end-to-end requirement

Input: $G\ V\ E\quad_{u\,v}\ F_{u\,v}\ x\ ,\quad 1,\mathrm{T}\quad 1\ src,m$
  repeat
    A find tree with HNN G T
    if A is found **then**
      decrease
    else
      increase
    end if
  until no significant increase in performance
Output: A is the multicast tree between *src* and *m*

The procedure transforms the random link descriptors into deterministic ones by using results from large deviation theory, which I formulated at (2.61).

$$\tilde{\mathrm{A}}_2 : \underset{\mathrm{A}}{\operatorname{argmin}} \quad C_{uv} \qquad \text{(2.61 revisited)}$$
$$s.t. \quad {}_{R_{src\,m}}\ s \quad \ln \quad s\ \mathrm{T}$$

The transformed problem can be seen as a CGSMT, which is still NP-hard, but I propose a sub-optimal solution by using HNN, where the corresponding parameters are described at subsubsection 2.4.3 and summarized in (2.78).

$$\mathbf{y} \quad {}_1\ 2\mathbf{y}^{tr}\mathbf{b}^{\,1} \quad {}_2\ \mathbf{y}^{tr}\mathbf{W}^{\,2}\,\mathbf{y} \quad 2\mathbf{y}^{tr}\mathbf{b}^{\,2} \qquad \text{(2.78 revisited)}$$

THESIS I.4 (optimizing link scaling using MAP/M/1)**.** *In* (2.100)*, I formulated a constrained optimization problem which connects the information about the random link descriptors (Link Entropy) and the appropriate bandwidth of the signaling process to support that information (Signaling Entropy) at a certain probability.*

$$\min_{t} \mathrm{H}\quad_{u\,v}\quad \mathbf{D0}\ \mathbf{D1}\quad {}_{u\,v}\quad \mathbf{D0}\ \mathbf{D1}\quad t \qquad \text{(2.100 revisited)}$$
$$s.t.\,\mathrm{H}\quad_{u\,v}\quad \mathbf{D0}\ \mathbf{D1}\quad t\quad {}_{u\,v}\quad t$$

I proposed a computable solution to this problem by modeling the dynamics of the link descriptors as MAP/M/1 described in (2.98) *and* (2.99)*,*

$$\mathrm{H}\quad_{u\,v}\quad \mathbf{D0}\ \mathbf{D1}\quad t\quad {}_{u\,v}\quad t \qquad \text{(2.98 revisited)}$$
$$\mathrm{H}\quad_{u\,v}\quad \mathbf{D0}\ \mathbf{D1}\quad {}_{u\,v}\quad \mathbf{D0}\ \mathbf{D1}\quad t \qquad \text{(2.99 revisited)}$$

Consequently the information theoretical quantities can be obtained analytically and the optimal solution can be found.

DOI:10.15774/PPKE.ITK.2019.006

Thesis group II - a heuristic solver based on hypergraphs for UBQP and its applicability in ICT

THESIS II.1 (A heuristic solver family based on hypergraphs for UBQP)**.** *In Algorithm 4 (restating), I have given a hypergraph based, easily parallelizable algorithm family to sub-optimally solve the UBQP problem.*

Algorithm 4 Pseudo code of the general UBQP solver algorithm

```
1: function INNER_SOLVER(W  k  k  b  k  y init  1  k)
2:     an arbitrary UBQP minimizer
3:     return y  1  k
4: end function
5: function (u  V_H  y  u_V)
6:     choose u  V_H                          choose the next hypernode and
7:     choose y  u_V                          choose a state in that hypernode
8:     return u  y
9: end function

Input: W b and u init                         the problem and the starting hypernode
10: u  u init  V_H                            start hypernode of the alg
11: choose y  u init  V                       init state in the hypernode
12: repeat
13:     define L W b y  objective function
14:     u  u and y  y
15:     W b  parameters from u  G V E Q W b
16:     if SHOULD_EMPLOY_INNER_SOLVER( )then
17:         y  INNER_SOLVER(W b y)
18:     else
19:         y  y
20:     end if
21:     u y  (u y )
22: until STOP_CRIT( )
Output: y                                     the best solution found by the alg.
```

The algorithms project the original search space into a hypergraph representation and use a HNN based internal solver to find a solution. I have given four instances of which two employs dimension reduction and two dimension addition. Table 3 (restating) summarizes the operation modes of the instances. (The precise description of the algorithms can be found in Appendix F)

Table 3 Categorization of the algorithms

	greedy	opportunistic
dim. reducer	L01	D01
dim. adder	DA02	DA01

I have tested the performance on three different problem sets: on the standard ORLIB UBQP benchmark set (subsection 3.4), on a scheduling problem (subsection 3.5), and on a simulated MUD problem (subsection 3.6). I have shown that the proposed methods perform near optimal on the investigated ICT problems.

DOI:10.15774/PPKE.ITK.2019.006

Thesis group III - near Bayesian performance non-parametric detection with Feed Forward Neural Networks

THESIS III.1 (blind detection by interval halving and FFNN)**.** *I have defined an FFNN based blind detector for the MUD problem, which lends itself to easy parallelization and can perform*

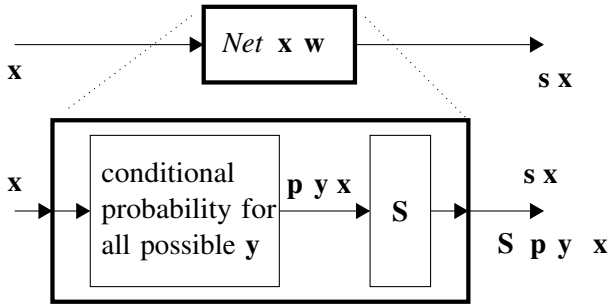


Figure 54: Equivalence of the FFNN with an encoding

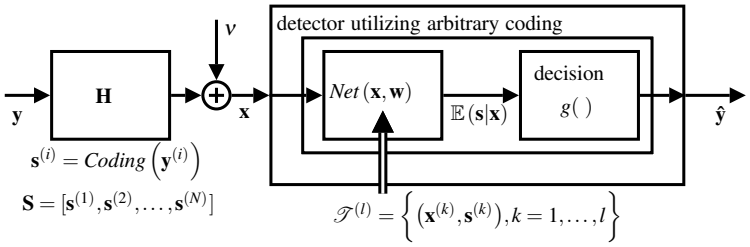


Figure 55: Flow graph representation of the detector using an arbitrary encoding

optimally under the constraint defined in (4.17)*.*

$$j \quad \max_{i \ 1 \ N} p\ \mathbf{s}^{\,i}\ \mathbf{x}$$
$$\text{if } j \quad E \qquad\qquad \text{(4.17 revisited)}$$
$$p\ \mathbf{s}^{\,k}\ \mathbf{x} \qquad\qquad p\ \mathbf{s}^{\,i}\ \mathbf{x}$$
$$k \ E \qquad\qquad i \ 1 \ N \ E$$

In (4.18)*, I give the linear encoding based on interval halving which is used to generate a training set for an FFNN*

$$S_{i\,j} \quad s_i^{\,j} \quad \text{sgn} \quad \sin \ 2 \ \ 2^{\,i\ 1}\ \frac{j}{N \ 1} \qquad i \ 1 \ L \ j \ 1 \ N \qquad \text{(4.18 revisited)}$$

and in (4.19) *I give the low complexity decision function which is to be employed on the output of the net.*

$$\hat{\mathbf{y}} \quad \text{sgn} \quad \mathbf{s}\ \mathbf{x} \qquad \text{(4.19 revisited)}$$

I have shown that the detector performs near optimally on the investigated MUD scenarios described in subsection 4.4.

DOI:10.15774/PPKE.ITK.2019.006

B Artificial Neural Networks outline

Ever since it was realized that our nervous system use neurons for computation there has been an interest to mimic that process and leverage the immerse efficiency. The first big milestone in this journey was laid down by Warren McCulloch and Walter Pitts in 1943 by the introduction of the first “artificial neuron” the TLU[113]. This model tries to mimic real life neurons with the crude simplification that they gather stimuli on their dendrites and if a threshold is exceeded an action potential is being fired on its axon. Most type of artificial neural networks use this or some variant of this simple processing unit. Although the literature extensively use the phrase “neural network” and “artificial neurons”, we know that these models are crude oversimplifications of the real biological units. My personal perspective which is based on the rapidly developing understanding of these biological systems[50, 153] is that the term “neural" should not be used on these units, but due to historical reasons I will refer to them as such. Nevertheless even these simple processing units can carry out vastly complex tasks if connected in a network. They are highly versatile, therefore are used in various engineering problems such as speech or pattern recognition, classification or data mining. Recent advances in “deep learning”[52, 73, 72] furthermore raised the interest of the field.

In this dissertation I employ two types of neural networks, namely Hopfield Neural Network (HNN) and Feed Forward Neural Network (FFNN). These networks are well understood and it is assumed that the reader has some basic knowledge[63, 64] in this field. Therefore I am summarizing only the relevant theorems and facts which are used to draw the conclusions of this dissertation.

B.1 Hopfield Neural Network (HNN)

Throughout the dissertation the Hopfield Neural Network is used as one main type of RNN. These networks are useful because of their inherent dynamics, massive parallelization capability and ease of representation. The dynamics by which the simplest type of HNNs operate can be summarized as follows: the network eventually arrives at one of its fix points which are determined by the local extrema of its energy function. This energy function is a quadratic function of the network’s state variables $\mathbf{y}$ and parametrized by $\mathbf{W}$ $\mathbf{b}$. In case of a Discrete Hopfield Neural Network the energy function can be described by the following set of equations:

$$\mathbf{y}\ \mathbf{W}\ \mathbf{b}\ :\quad \mathbf{y}^T\mathbf{W}\mathbf{y}\quad 2\mathbf{y}^T\mathbf{b} \tag{B.1a}$$

$$\mathbf{y}\qquad 1\ ^N\ \mathbf{b}\qquad ^N\ \mathbf{W}\qquad ^N\ ^N \tag{B.1b}$$

The dynamics of the network can be exploited if one can reformulate a task as an optimization problem where the solution lies at the extremum of such function. For example in the general binary case this problem is called the UBQP and it is proved to be NP-hard[45]:

$$\mathbf{y}_{opt}\qquad \min_{\mathbf{y}\quad 1\ ^N}\qquad \mathbf{y}\ \mathbf{W}\ \mathbf{b} \tag{B.2}$$

DOI:10.15774/PPKE.ITK.2019.006

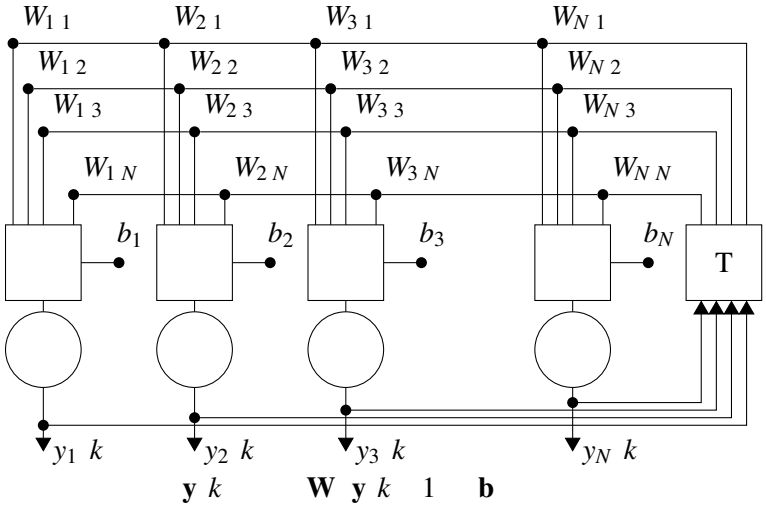


Figure 56: Block diagram of a DHNN

This simple type of network can also be thought as a 1-opt type local search algorithm. The fixed point in which the network settles will be largely determined by the initial state of the network, consequently greatly effecting the quality of the proposed solution. Several techniques were introduced (like randomization, applying hysteresis, etc) to overcome this phenomenon and can be applied to different tasks with various success rates.

Nevertheless there is an additional trait that can be exploited when using these networks, namely that the functional units are independent of each other and can operate in a parallel fashion. This makes it an ideal candidate for architectures that are based on computationally light but massively parallel execution.

B.2 Feed Forward Neural Network (FFNN)

FFNNs on the other hand have their general nonlinear approximation capabilities [25, 42, 76, 119, 121, 155] and also their inherent parallel architecture and trainability as traits.

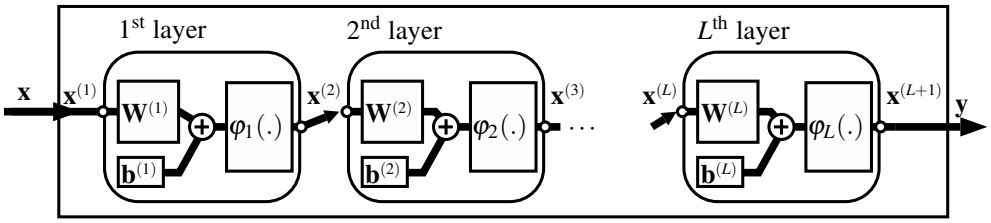


Figure 57: The general architecture of an FFNN

On Figure 57 one can see the general architecture of an FFNN. Note that $dim\ \mathbf{W}^i$ depends on the input and output size of the i^{th} layer. Also in general the $_i$ nonlinear functions can differ on any layer. The output of layer i can be described as:

$$\mathbf{x}^{i\ 1} \quad _i\ \mathbf{W}^i\ \mathbf{x}^i \quad \mathbf{b}^i \tag{B.3}$$

Also note that the "bias vector" is subtracted, but on the figure it is denoted with an addition for

convenience and conformance to literature.

FFNNs are exceptionally useful for detecting structures in raw data. This property comes from the fact that they can be "trained" to mimic a set of input-output pairs. When training an FFNN on a training set with l elements ^{l} $\mathbf{x}\,k$ $\mathbf{s}\,k$ k 1 l then the optimal weight vector is obtained by minimizing the following error function:

$$\mathbf{w}_{opt}^{l} : \min_{\mathbf{w}} \sum_{k\ 1}^{l} Net\ \mathbf{x}^{k}\ \mathbf{w}\ \mathbf{s}^{k\ 2} \qquad \text{(4.8 revisited)}$$

If the sample size tends to infinity, the FFNN asymptotically approximates the conditional expected value [63, 100]:

$$\lim_{l} Net\ \mathbf{x}\ \mathbf{w}_{opt}^{l}\ \mathbf{s}\ \mathbf{x}^{2}\ 0 \qquad \text{(4.9 revisited)}$$

Note that $\mathbf{s}$ in the training set can be arbitrarily chosen. Although many training algorithms exist, much of them can be categorized as a gradient-based learning methods. The most widely used base algorithm is the infamous backpropagation algorithm. A good overview on it and practical advises can be found in [98].

B.3 An outline of deep learning - perspective for neural networks

Around 2006 a new type of fast machine learning algorithm[72] started to gain focus, called "deep learning". This new type of algorithm and the corresponding neural network got the attention of the research community, because it was applied successfully in several practical and interesting cases[26, 73, 143, 51, 32], most notably in visual and speech recognition (like Microsoft Cortana, Xbox, Skype Translator, Amazon Alexa, Google Now, Apple Siri, etc), pattern generation (e.g. Google's deep dream[156, 30, 29] or the "artistic style transfer for videos"[140, 77]) and data classification problems. Ever since "deep learning" has been characterized as a buzzword, or a rebranding of neural networks[57]. Most of the recent advances are due to results from Prof. Geoffrey E. Hinton who is the author and co-author of several foundational publications[49]. In fact the terms "Contrastive learning", "deep belief networks" and "deep learning" were coined by him. Some of the important papers, video lectures and a good tutorial can be found at [52, 73, 72, 17, 71, 70, 31].

A Deep Belief Network (DBN) can be viewed as a specialized combined form of both RNNs and FFNNs. In fact they are stacked RBMs and can be trained in a greedy manner. As a result they learn to extract a hierarchical representation of the training data. After the initial unsupervised learning they can be easily fine tuned with gradient descent or backpropagation type algorithms. RBMs are restricted versions of the general BMs, and as such are stochastic, generative counterparts of HNNs. By stacking RBMs they form a special type of stochastic FFNN which can be run in both directions, both to analyze or to generate data. More information on RBMs can be found in[152, 73, 69].

DOI:10.15774/PPKE.ITK.2019.006

C Notation and general assumptions

Throughout the dissertation the following mathematical notations and assumptions are used, which mostly align with the conventional mathematical notations, but at some points introduce extensions with the intention to abbreviate and improve the ease of reading.

C.1 Notation of graphs

- a node weighted graph is denoted by $g \quad G\ V\ E\ Q\ \mathbf{W}\ \mathbf{b}$, where V denotes the set of nodes, and E denotes the set of edges and $Q\ \mathbf{W}\ \mathbf{b}$ is a function with two parameters describing the weights of the nodes.
- g_V denotes the node set V of graph g.
- g_E denotes the edge set E of graph g.
- $g_{\mathbf{W}}$ denotes the parameter $\mathbf{W}$ of the weight function in graph g. Similarly $g_{\mathbf{b}}$ denotes parameter $\mathbf{b}$.

C.2 Notation of sets, ordered sets (series), matrices, and vectors

- A set of size K is denoted by $A \quad a^1\ a^2 \quad a^K$ where ordering is not defined.
- An ordered set (series) of size K is denoted by $A \quad a^{\,1}\ a^{\,2} \quad a^{\,K}$, where the ordering is defined by the indexing.
- $\mathbf{W}$ denotes a matrix of, $\mathbf{W} \quad {}^{N\ N}$
- $\mathbf{W}^{\,i}$ denotes the i^{th} element in an ordered set of matrices. E.g. $\mathbf{W}^{\,4}$
- $\mathbf{W}^s$ denotes a specific element in a set of matrices of which the index is unknown or irrelevant, where s is a symbol. E.g. $\mathbf{W}^{\dagger}$.
- $\mathbf{b}$ denotes a vector of $\mathbf{b} \quad {}^{N}$
- elements of the ordered and unordered vector sets are denoted similarly as the matrices.

C.3 Notation of subsets and elements of matrices and vectors

Throughout the dissertation the following notation is used to describe individual components and sub-parts of the matrices and vectors, where the sub-parts could be the reordered versions of the original row or column wise.

$$\mathbf{A} \quad {}^{n\ n}\ S \quad 1 \quad n \quad S \quad 2^S$$

$$\text{if } R\ C \quad S\ I \quad 1 \quad n \quad R \quad J \quad 1 \quad n \quad C \tag{C.1a}$$

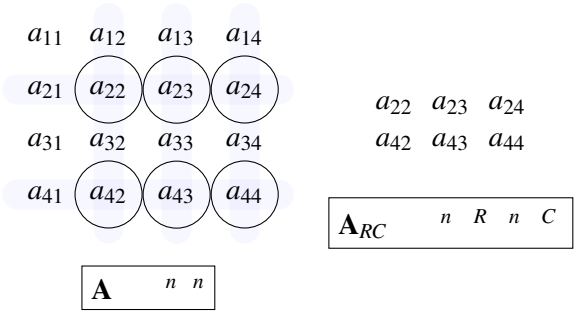
$$\text{then } \mathbf{A}_{RC} \quad {}^{n\ R\ n\ C} \tag{C.1b}$$

$$\text{s.t. } \mathbf{A}_{RC}\ i\ j \quad \mathbf{A}\ k\ l \quad k \quad R_{i\ i\ I}\ l \quad C_{j\ j\ J}$$

$$\text{if } R \quad C \text{ then } \mathbf{A}_{RC} : \quad \mathbf{A}_R \tag{C.1c}$$

Describes sub-matrix of a matrix of which original indices are $1 \quad n$. We apply the index series R and C to the rows and columns respectively. $\mathbf{A}_{RC}$ $\mathbf{A}$ then contains all row and columns in the order that are marked by the index series. For example if $n \quad 4 \quad \mathbf{A} \quad {}^{4\ 4}$, $R \quad 2\ 4$ and $C \quad 2\ 3\ 4$ then $\mathbf{A}_{RC} \quad {}^{2\ 3}$ and it contains the corresponding elements from rows $2\ 4$ and columns $2\ 3\ 4$

DOI:10.15774/PPKE.ITK.2019.006



of the original matrix $\mathbf{A}$. (C.1)

C.4 General assumptions on UBQP problems

These assumptions can be used without losing the generality of the UBQP, but nevertheless applied to make a common platform to evaluate and compare the results with the different formulations in different works.

- In the literature values of the variables in the UBQP are defined as 0 1 or 1 , but they can be generalized to any two distinct numbers by the following transformation[68]:

$$0\ 1 \leftrightharpoons F\ T \ : \tag{C.2a}$$

$$T01\_2\_FT\ \mathbf{y} \quad 0\ 1 \quad F\ T \ : \quad T \quad F \quad \mathbf{y} \quad F \tag{C.2b}$$

$$TFT\_2\_01\ \mathbf{y} \quad F\ T \quad F\ T \ : \quad \mathbf{y} \quad F \quad T \quad F \tag{C.2c}$$

- It is also known that a problem of dimension N containing a linear term can be transformed into a problem of dimension N 1 without the linear term by adding an additional constraint[68, 8, 28]. This process is called homogenization in the literature.
- Also for every UBQP we can say that $\mathbf{W}$ $\mathbf{W}^T$ aka symmetric. If we start from a problem formed with an asymmetric $\mathbf{W}$ there exist a simple transform which results in a symmetric parameter, but does not change neither the value nor the place of the extrema[12]: $\hat{\mathbf{W}}$ $\frac{1}{2}$ $\mathbf{W}$ $\mathbf{W}^T$
- The diagonal terms in $\mathbf{W}$ can be 0-ed out. If we have the variables y_i 0 1 then the diagonal terms can be rewritten as linear terms, because: $W_{ii}y_i^2$ $W_{ii}y_i$.
  In case of y_i 1 the diagonal terms can be left out since they become independent of the variables: $W_{ii}y_i^2$ W_{ii} 1.

DOI:10.15774/PPKE.ITK.2019.006

D A word on the log-moment generating function

The logarithm of the moment generating function (log-moment generating function) $\psi_X(s) = \ln M_X(s) = \ln \mathbb{E}[\exp(sX)]$ is an elementary tool when applying the large deviation theory and in particular the Chernoff bound. In this appendix I will show some used properties of it.

- $\psi_X(0) = 0$
- $M_X(s)$ is a logarithmically convex function or "superconvex" for any X random variable, thus $\psi_X(s)$ is also convex.
- $\psi_X(s)$ is a strictly monotone increasing function for any X random variable, if $X \neq 0$
- $\frac{d}{ds}\psi_X(s)$ exists and also strictly increasing for $X \neq 0$.
- Because of the previous property $\left(\frac{d}{ds}\psi_X(s)\right)^{-1}$ also exist and it is also strictly monotone increasing, if $X \neq 0$.
- $\sum_i \psi_{X_i}(s)$ has the same properties, since we are summing strictly monotone increasing functions.

Remark 4. *Note that in my discussion I only model properties (with random variables) which can take up positive values (like delay or energy consumption), so in these cases $X \neq 0$ always satisfied.*

I will demonstrate the strictly increasing property for a finite support discrete case, but a similar argument can be constructed for all other nondegenerate cases. The log-moment generating function for a discrete random variable is defined as:

$$\psi_X(s) = \ln \sum_{i=1}^{K} \exp(sx_i)\, p_i \tag{D.1}$$

Suppose that the support of this random variable is finite ($p_i \neq 0 \ \forall i \in [1, K]$) and the values are ordered from smallest to largest ($x_i \geq x_{i-1} \ \forall i \in [2, K]$).

The following expression which is used to find the optimal $\hat{s}$ parameter has a minimum and it can be found via a derivative.

$$\hat{s} = \inf_{s} \left\{ \psi_{X_j}(s) \leq sT \right\} \tag{D.2}$$

$$\frac{d\left(\psi_{X_j}(s) - sT\right)}{ds} = 0 \tag{D.3}$$

Proof. One can investigate the asymptotic behavior of $\psi_X(s)$ at $s = 0$ and $s = \infty$, by factoring out the largest term x_K and similarly the smallest x_1 since they will be the dominant ones:

$$\frac{d\,\psi_X(s)}{ds} = \frac{d \ln \sum_{i=1}^{K} \exp(sx_i)\, p_i}{ds} = \frac{\sum_{i=1}^{K} x_i \exp(sx_i)\, p_i}{\sum_{i=1}^{K} \exp(sx_i)\, p_i} = \sum_{i=1}^{K} \frac{x_i \exp(sx_i)\, p_i}{\sum_{k=1}^{K} \exp(sx_k)\, p_k} \tag{D.4}$$

$$= \frac{x_1}{1 + \sum_{k=2}^{K} \exp(s(x_k - x_1))\, p_k / p_1} + \sum_{i=2}^{K} \frac{x_i \exp(s(x_i - x_1))\, p_i / p_1}{1 + \sum_{k=2}^{K} \exp(s(x_k - x_1))\, p_k / p_1} \tag{D.5}$$

$$= \sum_{i=1}^{K-1} \frac{x_i \exp(s(x_i - x_K))\, p_i / p_K}{1 + \sum_{k=1}^{K-1} \exp(s(x_k - x_K))\, p_k / p_K} + \frac{x_K}{1 + \sum_{k=1}^{K-1} \exp(s(x_k - x_K))\, p_k / p_K} \tag{D.6}$$

DOI:10.15774/PPKE.ITK.2019.006

One can easily see that

$$\lim_s \frac{d\ X\ s}{ds} \quad \lim_s \frac{x_1}{1 \quad \sum_k^K {}_2 \exp\ s\ x_k \quad x_1 \quad \frac{p_k}{p_1}} \quad \sum_i^K {}_2 \frac{x_i \exp\ s\ x_i \quad x_1 \quad \frac{p_i}{p_1}}{1 \quad \sum_k^K {}_2 \exp\ s\ x_k \quad x_1 \quad \frac{p_k}{p_1}} \quad x_1 \tag{D.7}$$

$$\lim_s \frac{d\ X\ s}{ds} \quad \lim_s \sum_i^K {}_1 \frac{x_i \exp\ s\ x_i \quad x_K \quad \frac{p_i}{p_K}}{\sum_k^K {}_1 \exp\ s\ x_k \quad x_K \quad \frac{p_k}{p_K} \quad 1} \quad \frac{x_K}{\sum_k^K {}_1 \exp\ s\ x_k \quad x_K \quad \frac{p_k}{p_K} \quad 1} \quad x_K \tag{D.8}$$

since all exponential terms converge to 0 as s for (D.7) and when s for (D.8), because $x_i \quad x_1 \quad 0$ $i \quad 1$ and $x_i \quad x_K \quad 0$ $i \quad K$. Consequently

$$\lim_s \quad X\ s \quad sx_1 \quad c_1 \tag{D.9}$$

$$\lim_s \quad X\ s \quad sx_K \quad c_K \tag{D.10}$$

This means that asymptoticaly $X\ s \quad sT$ when s is approaching from acts as a linear function with slope $x_1 \quad T$ and when s is approaching it acts like a linear function with slope $x_K \quad T$. If $T \quad x_K$ the derivative is negative for every s, which means that the QoS parameter cannot be not satisfied, so an arbitrary s can be chosen. Likewise if $T \quad x_1$ the derivative is positive for every s, which means that the QoS parameter is impossible to satisfy, thus any s is an equally bad choice. If the QoS parameter $x_1 \quad T \quad x_K$ the slope of the derivative is $x_1 \quad T \quad 0$ for small s and $x_K \quad T \quad 0$ for large s. Consequently since the slope starts negative and becomes positive, somewhere inbetween has to be a minimum point. An example can be seen in Figure 58 where the support of X is $x_i \quad 1 \quad 10$ and the probabilities p_i were randomly generated.

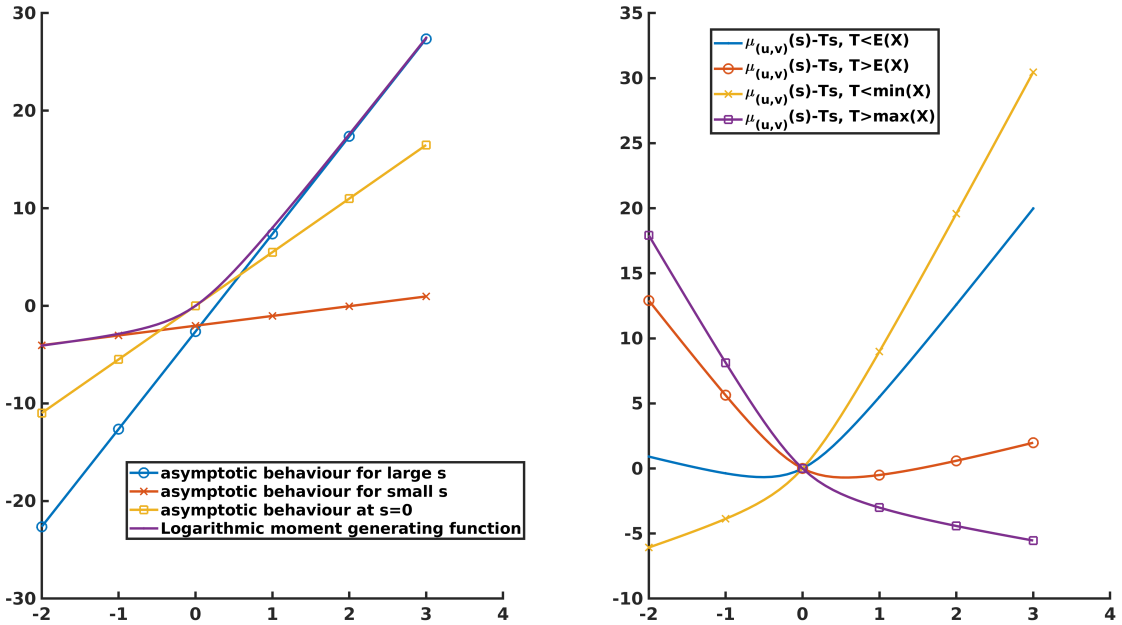


Figure 58: Limits for the log-moment generating function and existence of optimal s parameter

DOI:10.15774/PPKE.ITK.2019.006

Further investigating the limit (D.4) at $s \to 0$ we get:

$$\lim_{s \to 0} \frac{\sum_{i=1}^{K} x_i \exp(s x_i) p_i}{\sum_{i=1}^{K} \exp(s x_i) p_i} = \sum_{i=1}^{K} x_i p_i = E(X) \tag{D.11}$$

Based on this one can also state that the minimum point will be located at some $s > 0$ if $T > E(X)$ and located at $s < 0$ if $T < E(X)$.

The same argument can be derived for a sum of multiple log-moment generating functions, since the differentiation is a linear operator. The asymptotic behavior of the composite function for large s will be a linear function with slope $x_{1,K} + x_{2,K} + \dots + x_{N,K} - T$ and the fractional term becomes the sum of the individual fractional terms, which also vanishes. For negative s the asymptotic behavior is the same, with slope $x_{1,1} + x_{2,1} + \dots + x_{N,1} - T$. Also note that every $\mu_X(s)$ for any X is a strictly monotone increasing function, so their sum is also strictly monotone increasing. This means that $\mu_X(s) - sT$ can have at most one inflexion point. If it has, it is a minimum. An important remark is that using the Chernoff bound, parameter s can take only positive values. □

Also the Chernoff bound essentially cannot give anything "meaningful" left to the original random variable's mean value, because $\lim_{s \to 0} d\mu_X(s)/ds = E(X)$ and $\exp(\mu_X(0)) = 1$. It is well known that the Chernoff bound is sharper for random variables with "heavy heads" meaning that the mean is located "more to the left". The following figures depict the sharpest possible Chernoff bounds for a Geometric distribution and a Poisson distribution. Note that left to the mean values the Chernoff bound gives probability 1.

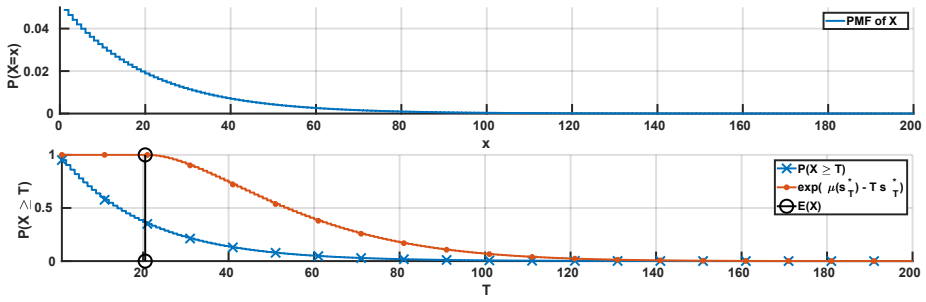


Figure 59: Sharpest Chernoff bounds for a Geometric distribution

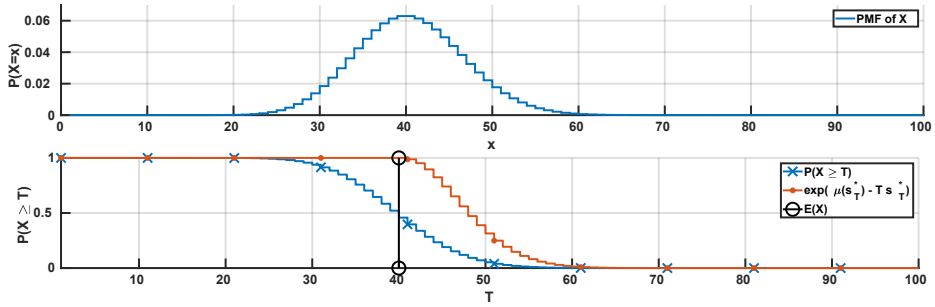


Figure 60: Sharpest Chernoff bounds for a Poisson distribution

DOI:10.15774/PPKE.ITK.2019.006

E Short description of the GSMT and CGSMT problem

The general SMT problem can be viewed as a generalization of the infamous Fermat problem[24]. A survey written in 1992 regarding Steiner trees can be found at [81]. Another survey[138] from 2009, which summarizes the used heuristics and methods for finding an SMT. The general SMT problem can be defined as follows: We have a set M containing the target points. One has to find a set S containing the Steiner points, such that the spanning tree on $M \quad M$ should be of minimal weight (total length of edges). This tree is called the SMT or in short Steiner tree

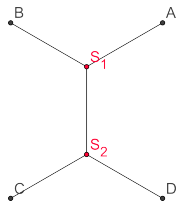


Figure 61: A Steiner tree of 4 points

E.1 The GSMT problem

The GSMT problem differs from the SMT such that the problem is defined on a graph G, and the points of S cannot be chosen arbitrarily but from the existing vertices of G. On this edge weighted graph $G \quad V \; E$, we have a set $M \quad V$ which must be contained in the final tree. We search for a set $S \quad V$ for which the tree $T \quad S \quad M \; E_T$ has minimal weight. Note that T need not be a MST, thus any potential graph point can be a Steiner point and any other can be left out from the tree. Let us denote the set of all trees on $G \quad V \; E \; W$ by , where W denotes the weights of the edges. T denotes a tree of $T \quad V_T \; E_T \; W_T$, then the GSMT problem can be formulated as

$$\begin{aligned} T_{opt} \quad & \underset{T}{\operatorname{argmin}} \quad \underset{u\,v \quad E_T}{W_{u\,v}} \\ & \text{s.t. } M \quad V_T \end{aligned} \tag{E.1}$$

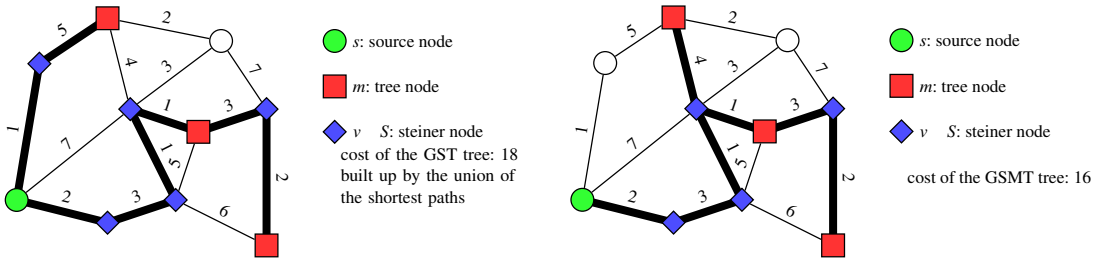
where is a general objective function which we want to minimize. Usually the objective function is just the sum of the edge weights. Figure 62 depicts two examples for possible trees in the graph. Figure 62a was built by taking the union of all shortest paths from s to all $m \quad M$. This naiive construction does not results in a GSMT.

E.2 The CGSMT problem

The CGSMT is the extension of the GSMT problem, where an additional constraint is posed which the tree must satisfy.

$$\begin{aligned} T_{opt} \quad & \underset{T}{\operatorname{argmin}} \quad \underset{u\,v \quad E_T}{W_{u\,v}} \\ & \text{s.t.} \quad M \quad V_T \text{ and } \underset{u\,v \quad E_T}{\qquad} u \; v \quad \text{is TRUE} \end{aligned} \tag{E.2}$$

DOI:10.15774/PPKE.ITK.2019.006



(a) A tree built from the union of shortest paths from s to all m. Note that this is not a GSMT

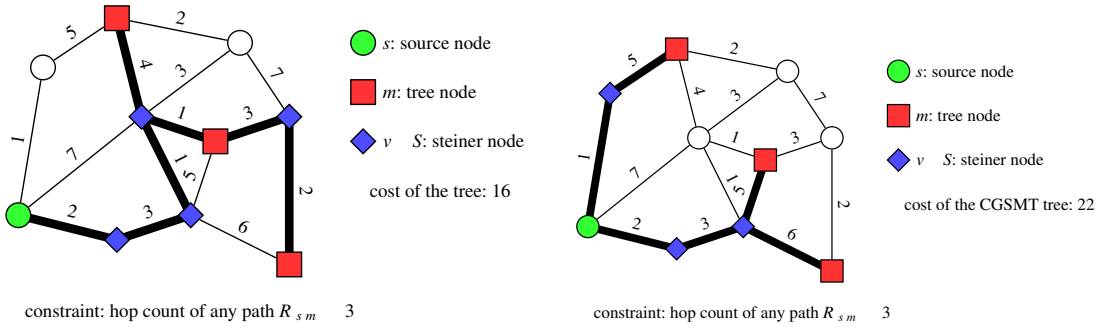
(b) A GSMT on the same graph

Figure 62: Example of two trees on the graph.

A typical objective function is where we want to minimize the weight of the tree which builds up from summing the weight of the tree edges. But at the same time a constraint is posed that any route in the tree from s to any m must not exceed a quality parameter along the route.

$$T_{opt} \quad \underset{T}{\text{argmin}} \sum_{u\,v \quad E_T} W_{u\,v}$$
$$\text{s.t.} \quad N \quad V_T \text{ and } \sum_{u\,v \quad R_{s\,m}} D_{u\,v} \quad m \quad M \tag{E.3}$$

In the following easy example the weights of the edges are summed up that we want to minimize the total weight of the tree, but at the same time the constraint which the tree must satisfy is such that along every route the hop count must be less than 3. Figure 63 shows two sub-figures. The first depicts the unconstrained GSMT but at the same time it is an invalid CGSMT, because it does not satisfy the posed constraint. The second depicts the correct CGSMT.



(a) Incorrect CGSMT, where the constraint is not met

(b) Correct CGSMT

Figure 63: An example for an incorrect and a correct CGSMT

DOI:10.15774/PPKE.ITK.2019.006

F Description and performance characterization of the new UBQP related algorithms

In this appendix we give the precise definition of the algorithms used in this article. We define the rules presented in subsection 3.3.

F.1 Dimension reduction of the greedy algorithm - L01

This algorithm uses a greedy strategy at the hypergraph level to determine in which direction the next hypernode falls. The algorithm stores the best N dimensional candidate solution in $\mathbf{y}^{\mathbf{opt}}$.

1. The algorithm starts at the original N dimensional hypernode.
2. The inner solver is chosen to be a DHNN structure.
3. The performance of a "candidate solution" is defined by the value of the N dimensional quadratic function on this solution.
4. We use the "inner solver" in every hypernode.
5. The next $n \quad 1$ dimensional hypernode is chosen from an $n \quad N$ dimensional hypernode as follows ( ): We search all the possible $n \quad 1$ dimensional hypernodes accessible from the current hypernode, and pick the best among them, which is performed by the following steps:
   - We assume that if we are in an arbitrary $n \quad N$ dimensional hypernode the best N dimensional candidate solutions is accessible in $\mathbf{y}^{opt}$.
   - We select a starting point in an $n \quad 1$ dimensional hypernode for the "inner solver" from the actual $\mathbf{y}^{opt}$ by discarding the appropriate dimensions.
   - In each $n \quad 1$ dimensional hypernode the "inner solver" generates a $n \quad 1$ dimensional candidate solution, we denote it with $\mathbf{y}^{\ i}$ next .
   - We map every $\mathbf{y}^{\ i}$ next back to the original N dimensional space by filling the missing coordinates of $\mathbf{y}^{\dagger\ i}$ next with the coordinates of $\mathbf{y}^{opt}$.
   - According to the N dimensional quadratic function we pick the best $\mathbf{y}^{\dagger\ i}$ next and compare with the value of the quadratic function taken over $\mathbf{y}^{opt}$.
   - If the performance is improved we choose that hypernode.
6. If not, the algorithm stops.

F.2 Dimension reduction of the first chance algorithm - D01

This algorithm is different from the one described above in one rule. We do not evaluate all possible lower dimensional hypernodes, but if we find one which gives a better candidate solution then we choose that hypernode. So instead of a full evaluation we introduce a "first-improve" strategy.

1. The algorithm starts at the original N dimensional hypernode.
2. The inner solver is chosen to be a DHNN structure.
3. The performance of a "candidate solution" is defined by the value of the N dimensional quadratic function on this solution.
4. We use the "inner solver" in every hypernode.

DOI:10.15774/PPKE.ITK.2019.006

5. The next hypernode from an n N dimensional hypernode is chosen as follows ( ). We pick one by one a possible n 1 dimensional hypernode from the current node and if it improves our current candidate solution, we choose that hypernode, which is performed by the following steps:
   - We assume that if we are in an arbitrary n N dimensional hypernode the best N dimensional candidate solutions is accessible in $\mathbf{y}^{opt}$.
   - We select a starting point in an n 1 dimensional hypernode for the "inner solver" from the actual $\mathbf{y}^{opt}$ by discarding the appropriate dimensions.
   - In that n 1 dimensional hypernode the "inner solver" generates one specific n 1 dimensional candidate solution, we denote it by $\mathbf{y}^{\ i}$ next .
   - We map $\mathbf{y}^{\ i}$ next back to the original N dimensional space by filling the missing coordinates of $\mathbf{y}^{\dagger\ i}$ next with the coordinates of $\mathbf{y}^{opt}$.
   - According to the N dimensional quadratic function we compare them with each other.
   - If the performance is improved we choose that hypernode.
   - If not, we try another not yet inspected hypernode in the n 1 dimensional regime.
6. If we inspected every possible n 1 dimensional hypernode and did not find a better candidate solution, the algorithm stops.

F.3 The description of dimension adder DA01 algorithm

This algorithm constructs a candidate solution gradually by adding dimensions starting from a low dimensional hypernode until it reaches the highest dimensional hypernode.

1. The algorithm starts from the 0 dimensional hypernode.
2. The inner solver is chosen to be a DHNN structure.
3. The performance of a candidate solution depends on the dimension as we leave out the corresponding parts of the matrix and vector of the original N dimensional quadratic function
4. We use the "inner solver" in every hypernode.
5. The next hypernode from an n N dimensional hypernode is chosen as follows ( ): We pick one by one a possible n 1 dimensional hypernode from the current node and if it improves our current candidate solution, we choose that hypernode, which is performed by the following steps:
   - We denote the set containing the indices of the coordinates of the current n dimensional hypernode with C. We denote the proposed candidate solution in this hypernode with $\mathbf{y}$.
   - We choose randomly a dimension which has not yet been picked, denoted by: B i i C.
   - We inspect the n 1 dimensional hypernode with dimension indices A C B with the "inner solver".
   - At the inspected n 1 dimensional hypernode the starting point of the "inner solver" (denoted by $\mathbf{y}$) is generated via copying the appropriate coordinates from the best found candidate solution and computing the missing coordinate via the gradient. $\mathbf{y}_C$ $\mathbf{y}$ and $\mathbf{y}_B$ sgn $\mathbf{W}^{\text{orig}}{}_{BC}$ $\mathbf{y}$ $\mathbf{b}^{\text{orig}}{}_B$

DOI:10.15774/PPKE.ITK.2019.006

- We use the “inner solver” to get $\mathbf{y}\quad next\quad$ from $\mathbf{y}$.

6. If the N^{th} dimension was also added to the problem we stop and put $\mathbf{y}^{opt}\quad\mathbf{y}\quad next\ .$

F.4 The description of the dimension adder DA02 algorithm

This algorithm is similar DA01 algorithm. It also constructs a candidate solution from a lower dimensional one, but instead of a first-improve hypernode choice strategy this inspects all the possible one distance higher dimensional hypernodes and chooses the best one.

1. The algorithm starts from the 0 dimensional hypernode.
2. The “inner solver” is a DHNN structure.
3. The performance of a candidate solution here is dynamic. It is determined by the current $n\quad 1\quad N$ dimensional quadratic function.
4. We use the “inner solver” in every hypernode.
5. The strategy by which we choose the next hypernode ( ) is the following:
   - We assume that if we are in an n dimensional hypernode, we also know the indices of the coordinates of the said hypernode. We denote this set with C, and the proposed candidate solution in this hypernode with $\mathbf{y}$.
   - We iterate through every dimension index which we did not inspect so far: $i : B\quad i\quad i\quad C$. We inspect all the possible $n\quad 1$ dimensional hypernode with dimension indices $A\quad C\quad B$ with the “inner solver”.
   - At each inspected $n\quad 1$ dimensional hypernode the starting point of the “inner solver” (denoted by $\mathbf{y}^{i}$) is generated via copying the appropriate coordinates from the best found candidate solution and computing the missing coordinate via the gradient. $\mathbf{y}^{i}_{C}\quad\mathbf{y}$ and $\mathbf{y}^{i}_{B}\quad\text{sgn}\ \mathbf{W}^{\text{orig}}{}_{B\,C}\ \mathbf{y}\quad\mathbf{b}^{\text{orig}}{}_{B}$
   - We use the “inner solver” to get $\mathbf{y}^{\ i}\quad next\quad$ from $\mathbf{y}^{\ i}$.
   - We choose the best performing $\mathbf{y}^{\ i}\quad next\quad$ and the corresponding hypernode for the next iteration.
6. If the N^{th} dimension was inspected as well the algorithm stops, and we put $\mathbf{y}^{opt}\quad\mathbf{y}\quad next\ .$

F.5 Run-time and performance analysis tables of the UBQP solvers

Table 4 Performance analysis table for the ORLIB problems

		mean solution											best solution										relative frequency of best solution									
	alg/prob	1-opt LS	BLS	BTS	D01	DA01	DA02	DDT	HNN	L01	SDR	SDR md	1-opt LS	BLS	BTS	D01	DA01	DA02	DDT	HNN	L01	SDR	SDR rnd	1-opt LS	BLS	BTS	D01	DA01	DA02	DDT	HNN	L01	SDR	SDR rnd
dim=50 K=1000	1	1907	1851,3	**2160**	2053,8	2056,7	2092	2142	1889,9	2134,7	1168	1168	**2160**	**2160**	**2160**	**2160**	**2160**	2092	2142	**2160**	**2160**	1168	1168	0,01	0,00	1,00	0,14	0,10	1,00	1,00	0,01	0,65	1,00	1,00
	2	3450,9	3427,7	**3658**	3553,4	3589,4	3102	3538	3310,3	3636,8	3092	3092	**3658**	**3658**	**3658**	**3658**	**3658**	3102	3538	**3658**	**3658**	3092	3092	0,14	0,17	1,00	0,22	0,14	1,00	1,00	0,04	0,94	1,00	1,00
	3	4624,18	4586,2	4650	4714,4	4680	4590	4594	4556,8	**4772,8**	4198	4198	**4778**	**4778**	4650	**4778**	**4778**	4590	4594	**4778**	**4778**	4198	4198	0,27	0,20	1,00	0,52	0,49	1,00	1,00	0,14	0,95	1,00	1,00
	4	3291,01	3311,3	**3472**	3397,6	3403,4	3020	**3472**	3319,4	3452,2	3144	3144	**3472**	**3472**	**3472**	**3472**	**3472**	3020	**3472**	**3472**	**3472**	3144	3144	0,09	0,03	1,00	0,14	0,13	1,00	1,00	0,05	0,63	1,00	1,00
	5	3943,51	3959,2	**4152**	4110,8	4102,7	**4152**	4030	3984,1	4114,1	3334	3334	**4152**	**4152**	**4152**	**4152**	**4152**	**4152**	4030	**4152**	**4152**	3334	3334	0,19	0,27	1,00	0,47	0,72	1,00	1,00	0,25	0,54	1,00	1,00
	6	3762,06	3764,2	**3842**	3841,7	3823,1	**3842**	3802	3813,5	**3842**	3412	3412	**3842**	**3842**	**3842**	**3842**	**3842**	**3842**	3802	**3842**	**3842**	3412	3412	0,42	0,29	1,00	1,00	0,53	1,00	1,00	0,33	1,00	1,00	1,00
	7	4482,53	4470,9	**4588**	4520,5	4535,3	4530	4578	4466,4	4572,6	3986	3986	**4588**	**4588**	**4588**	**4588**	**4588**	4530	4578	**4588**	**4588**	3986	3986	0,18	0,21	1,00	0,10	0,26	1,00	1,00	0,09	0,61	1,00	1,00
	8	4056,29	4049,7	**4222**	4212,2	4195,4	**4222**	4062	4128,3	4220,8	3698	3698	**4222**	**4222**	**4222**	**4222**	**4222**	**4222**	4062	**4222**	**4222**	3698	3698	0,30	0,29	1,00	0,80	0,85	1,00	1,00	0,64	0,95	1,00	1,00
	9	3680,42	3662,4	**3862**	3841,6	3827,5	3720	**3862**	3693,9	3855,5	3600	3600	**3862**	**3862**	**3862**	**3862**	**3862**	3720	**3862**	**3862**	**3862**	3600	3600	0,16	0,18	1,00	0,71	0,59	1,00	1,00	0,21	0,88	1,00	1,00
	10	3351,22	3291,4	**3496**	3375	3452,9	3460	**3496**	3273,1	3478,8	3332	3332	**3496**	**3496**	**3496**	**3496**	**3496**	3460	**3496**	**3496**	**3496**	3332	3332	0,09	0,03	1,00	0,02	0,23	1,00	1,00	0,02	0,53	1,00	1,00
	alg/prob	1-opt LS	BLS	BTS	D01	DA01	DA02	DDT	HNN	L01	SDR	SDR md	1-opt LS	BLS	BTS	D01	DA01	DA02	DDT	HNN	L01	SDR	SDR rnd	1-opt LS	BLS	BTS	D01	DA01	DA02	DDT	HNN	L01	SDR	SDR rnd
dim=100 K=100	1	7450,84	7459,3	**7910**	7692,2	7641,8	7558	7048	7456,6	7781,8	4430	4552	7812	7804	**7910**	**7910**	7858	7558	7048	7812	**7910**	4430	4552	0,01	0,01	1,00	0,03	0,03	1,00	1,00	0,01	0,06	1,00	1,00
	2	10916,2	10881	**11178**	11082	11021	11170	10904	10853	11136	9274	9274	**11178**	11170	**11178**	**11178**	**11178**	11170	10904	11170	**11178**	9274	9274	0,01	0,01	1,00	0,09	0,01	1,00	1,00	0,01	0,17	1,00	1,00
	3	12756,3	12726	**12956**	12902	12870	12898	12898	12780	12923	11476	11476	**12956**	**12956**	**12956**	**12956**	**12956**	12898	12898	**12956**	**12956**	11476	11476	0,08	0,09	1,00	0,30	0,24	1,00	1,00	0,09	0,40	1,00	1,00
	4	10304,9	10337	**10606**	10529	10482	10596	9856	10275	10578	8696	8724	**10606**	**10606**	**10606**	**10606**	**10606**	10596	9856	**10606**	**10606**	8696	8724	0,04	0,02	1,00	0,38	0,05	1,00	1,00	0,02	0,62	1,00	1,00
	5	8664,52	8621,2	**8994**	8814,2	8793,7	8738	8612	8621,4	8891,7	6332	6332	**8996**	8948	8994	**8996**	**8996**	8738	8612	**8996**	**8996**	6332	6332	0,01	0,01	1,00	0,16	0,02	1,00	1,00	0,01	0,34	1,00	1,00
	6	10167,1	10116	**10470**	10406	10370	10404	9962	10174	10458	8520	8520	**10486**	10456	10470	**10486**	**10486**	10404	9962	**10486**	**10486**	8520	8520	0,02	0,02	1,00	0,07	0,02	1,00	1,00	0,01	0,22	1,00	1,00
	7	9686,64	9772,2	**9980**	9908,6	9913	9798	9672	9756,4	9952,5	8360	8476	9998	**10030**	9980	**10030**	**10030**	9798	9672	9998	**10030**	8360	8476	0,02	0,01	1,00	0,01	0,03	1,00	1,00	0,04	0,05	1,00	1,00
	8	11072,8	11114	**11380**	11282	11252	11140	11220	11116	11322	9408	9408	**11380**	11364	**11380**	**11380**	**11380**	11140	11220	**11380**	**11380**	9408	9408	0,05	0,01	1,00	0,45	0,14	1,00	1,00	0,04	0,80	1,00	1,00
	9	11130,2	11047	**11340**	11234	11210	11098	11118	11089	11316	9646	9646	**11340**	**11340**	**11340**	**11340**	**11340**	11098	11118	**11340**	**11340**	9646	9646	0,24	0,15	1,00	0,36	0,28	1,00	1,00	0,13	0,70	1,00	1,00
	10	12266,6	12247	**12438**	12363	12339	12276	11922	12264	12407	10898	10898	12436	**12438**	**12438**	**12438**	**12438**	12276	11922	12436	**12438**	10898	10898	0,02	0,03	1,00	0,22	0,03	1,00	1,00	0,02	0,42	1,00	1,00
	alg/prob	1-opt LS	BLS	BTS	D01	DA01	DA02	DDT	HNN	L01	SDR	SDR md	1-opt LS	BLS	BTS	D01	DA01	DA02	DDT	HNN	L01	SDR	SDR rnd	1-opt LS	BLS	BTS	D01	DA01	DA02	DDT	HNN	L01	SDR	SDR rnd
dim=500 K=50	1	113350	113980	**116526**	115013	114786	115872	106718	113732	115390	80536	80536	115618	115942	**116526**	116352	116504	115872	106718	115682	116474	80536	80536	0,02	0,02	1,00	0,02	0,02	1,00	1,00	0,02	0,02	1,00	1,00
	2	126722	127514	**128678**	128234	127863	127076	118614	127232	128330	97186	97186	128196	128434	**128678**	**128678**	128572	127076	118614	128406	**128678**	97186	97186	0,02	0,02	1,00	0,04	0,02	1,00	1,00	0,02	0,10	1,00	1,00
	3	128309	129396	**131084**	130478	129982	129976	120526	129246	130823	99388	99388	130370	130740	**131084**	**131084**	131054	129976	120526	130616	**131084**	99388	99388	0,02	0,02	1,00	0,04	0,02	1,00	1,00	0,02	0,12	1,00	1,00
	4	127186	128241	**129794**	128938	128682	128210	120978	127733	129265	100970	100970	129008	129538	129794	129734	129764	128210	120978	129408	**129812**	100970	100970	0,02	0,02	1,00	0,02	0,02	1,00	1,00	0,02	0,02	1,00	1,00
	5	122659	123417	**125008**	124313	123845	124338	116426	123108	124639	88308	88308	125004	124750	125008	**125098**	124806	124338	116426	124978	**125098**	88308	88308	0,02	0,02	1,00	0,04	0,02	1,00	1,00	0,02	0,12	1,00	1,00
	6	119157	119295	**121868**	120503	120344	120410	112208	119234	120804	92758	92758	121332	121124	**121868**	121710	121630	120410	112208	120950	121816	92758	92758	0,02	0,02	1,00	0,02	0,02	1,00	1,00	0,02	0,02	1,00	1,00
	7	119791	120098	**122730**	121346	121306	120728	112264	120001	121790	88458	88608	121920	122012	**122730**	122588	122444	120728	112264	121722	**122730**	88458	88608	0,02	0,02	1,00	0,02	0,02	1,00	1,00	0,02	0,02	1,00	1,00
	8	120785	120955	**123454**	122353	121857	121810	113920	120478	122561	93652	93652	123042	123042	123454	123262	**123472**	121810	113920	122638	123436	93652	93652	0,02	0,02	1,00	0,02	0,02	1,00	1,00	0,02	0,02	1,00	1,00
	9	118318	119593	**121622**	120313	120139	119788	108248	118832	120824	86378	86378	120824	121378	121622	121420	121240	119788	108248	121130	**121636**	86378	86378	0,02	0,02	1,00	0,02	0,02	1,00	1,00	0,02	0,02	1,00	1,00
	10	128976	129410	**130900**	130541	130145	130684	123996	129550	130840	99076	99636	130370	131086	130900	**131374**	131238	130684	123996	131194	**131374**	99076	99636	0,02	0,02	1,00	0,06	0,06	1,00	1,00	0,02	0,06	1,00	1,00

DOI:10.15774/PPKE.ITK.2019.006

Table 5 Run time table for the ORLIB problems

		average relative time											std relative time										
	alg/prob	1-opt LS	BLS	BTS	D01	DA01	DA02	DDT	HNN	L01	SDR	SDR rnd	1-opt LS	BLS	BTS	D01	DA01	DA02	DDT	HNN	L01	SDR	SDR rnd
dim=50 K=1000	1	22,0	3,2	3822,6	38,5	14,3	81,0	758,6	1,0	413,9	1334,4	1434,7	3,85	0,73	156,08	10,66	1,38	4,72	15,88	0,32	58,81	49,36	112,49
	2	24,7	4,5	3845,2	41,6	14,4	82,5	693,4	1,0	438,7	1332,3	1318,8	4,15	1,02	138,32	10,60	1,73	2,20	23,18	0,20	57,29	76,02	86,23
	3	21,7	3,6	3007,7	26,9	11,0	65,0	573,3	1,0	347,5	1042,5	1026,6	3,50	0,71	139,09	7,46	0,83	2,00	24,05	0,16	42,57	82,77	59,15
	4	21,3	3,2	3342,0	33,0	12,4	72,0	568,1	1,0	393,5	1158,6	1076,4	3,27	0,60	127,82	9,04	1,24	1,56	6,31	0,31	51,61	82,12	78,57
	5	22,9	4,2	3501,3	30,2	12,5	72,6	605,7	1,0	352,3	1059,0	1058,0	3,46	0,82	140,13	7,19	0,40	1,86	12,25	0,15	9,93	10,58	22,61
	6	21,7	3,7	2997,6	28,0	10,7	66,1	486,5	1,0	281,4	923,2	928,0	3,43	0,73	126,22	6,60	0,46	2,86	19,09	0,13	4,24	18,24	32,83
	7	21,3	3,3	3163,8	26,1	11,7	67,3	587,7	1,0	324,1	1002,0	1017,9	3,14	0,64	195,21	3,99	0,44	1,28	28,73	0,13	5,60	29,47	54,86
	8	22,4	3,3	3031,0	26,1	11,3	65,1	597,3	1,0	308,6	934,1	930,1	3,81	0,69	117,68	5,23	0,42	3,13	3,96	0,24	7,22	13,36	9,85
	9	22,4	3,9	3302,5	34,9	11,9	70,4	622,9	1,0	342,9	992,4	996,7	3,52	0,78	142,31	10,07	0,48	2,30	4,25	0,14	9,89	8,88	26,04
	10	19,1	3,3	3078,5	28,7	11,3	66,6	508,5	1,0	312,2	887,3	879,9	3,12	0,65	129,80	7,90	0,45	2,02	5,03	0,12	8,77	19,20	8,85
	alg/prob	1-opt LS	BLS	BTS	D01	DA01	DA02	DDT	HNN	L01	SDR	SDR rnd	1-opt LS	BLS	BTS	D01	DA01	DA02	DDT	HNN	L01	SDR	SDR rnd
dim=100 K=100	1	76,5	9,6	2187,3	110,8	22,4	250,9	1290,6	1,0	1605,2	1059,7	1069,5	9,73	1,27	111,55	34,62	0,82	4,57	13,39	0,18	75,95	9,74	13,39
	2	86,0	9,6	2230,7	118,4	27,0	252,9	1288,6	1,0	1662,6	1104,8	1110,8	9,87	1,60	123,34	39,29	3,15	3,40	38,00	0,18	74,71	44,01	35,99
	3	91,6	9,2	2255,1	84,4	22,6	261,3	1365,8	1,0	1706,5	1157,2	1222,4	11,36	1,48	105,51	23,93	0,72	14,99	54,97	0,19	57,89	55,21	62,11
	4	92,5	10,4	2103,1	86,6	22,7	241,3	1288,1	1,0	1557,8	1136,2	1098,5	19,91	1,57	109,55	18,98	2,04	3,46	64,43	0,16	73,90	101,99	14,66
	5	82,4	8,3	2193,2	101,9	22,9	250,8	1236,2	1,0	1616,4	1098,0	1104,0	11,29	1,26	113,20	28,72	5,11	2,85	13,35	0,20	53,10	12,24	8,57
	6	79,0	8,2	2083,2	87,5	21,4	235,9	1319,6	1,0	1581,0	1047,0	1054,9	8,84	1,35	82,84	25,39	0,64	2,01	5,15	0,17	45,10	7,47	7,84
	7	87,0	9,0	2329,8	103,7	25,9	263,5	1385,1	1,0	1813,8	1124,6	1138,8	10,00	1,43	93,26	31,11	0,73	2,36	3,58	0,18	41,65	9,24	27,01
	8	82,5	10,4	2066,3	87,0	21,2	233,1	1233,8	1,0	1554,0	1069,2	1069,7	11,27	1,42	106,52	25,70	0,60	1,83	4,55	0,19	87,12	8,45	8,06
	9	82,4	8,1	2032,7	73,1	21,0	233,3	1291,9	1,0	1451,3	1056,5	1051,0	8,99	1,11	96,97	15,39	0,67	2,16	13,51	0,22	51,25	6,05	5,16
	10	72,5	7,4	1989,5	81,2	19,6	215,4	1135,5	1,0	1476,4	979,4	981,1	8,73	1,12	132,19	19,72	0,63	2,05	19,81	0,20	35,27	26,10	20,94
	alg/prob	1-opt LS	BLS	BTS	D01	DA01	DA02	DDT	HNN	L01	SDR	SDR rnd	1-opt LS	BLS	BTS	D01	DA01	DA02	DDT	HNN	L01	SDR	SDR rnd
dim=500 K=50	1	279,715	30,532	505,98	314,97	56,515	1969,1	8484,8	1	36157	3077,5	3003,668	24,9352	4,33	40,702	84,1	3,68	181	740	0,25	3328,6	141,7	263,5164
	2	264,855	26,897	698,82	277,55	50,724	1809,8	11003	1	32571	2856,4	2836,531	18,9195	2,01	27,285	104	2,48	20,1	177	0,31	3512	16,8	23,94702
	3	261,731	33,051	663,24	301,84	64,037	2322,7	12926	1	29893	3584,3	3744,132	25,2955	3,04	60,186	109	1,95	13,6	1788	0,33	1895,6	108,6	79,96405
	4	261,234	25,362	699,35	289,36	48,94	1736,3	11415	1	33865	2809,1	2764,547	21,3041	2,26	48,396	93,9	2,25	18	210	0,25	2200	26,92	20,32727
	5	283,293	32,782	673,46	319,47	71,037	2601,1	11441	1	34010	4090,2	3974,91	23,6563	2,95	31,945	98,5	3,46	62,2	1128	0,26	2836	132,4	48,58605
	6	270,502	26,707	770,49	348,37	51,658	1836	11728	1	31572	2922,2	2929,717	19,4891	2,2	46,972	132	2,77	18	251	0,31	2419,8	38,15	19,62122
	7	301,702	26,797	789,46	366,05	58,667	2101,5	12983	1	31455	3309	3309,102	21,5418	3,05	3,1439	116	2,55	20,5	202	0,24	3228	31,4	18,15639
	8	266,029	27,252	691,84	325,26	50,877	1873,8	11418	1	30789	2907,8	2889,149	19,5633	2,41	63,478	107	3,15	11,8	188	0,24	2817,3	34,76	21,72813
	9	285,154	29,503	636,52	322,62	53,894	1955,9	8568,5	1	37737	3209	2891,988	23,3678	3,81	1,8979	84	6,59	139	436	0,22	3369	197,7	281,7607
	10	264,858	30,58	665,72	290,61	62,966	2354	11394	1	31382	3813,4	3697,525	20,8994	2,6	14,487	84,7	1,97	117	1504	0,31	2338,2	119,4	22,01743

DOI:10.15774/PPKE.ITK.2019.006

DOI:10.15774/PPKE.ITK.2019.006

G List of figures, tables and algorithms

List of Figures

1 Graph model of the network . . . 8
2 LSA in case the link descriptor is of delay type. . . . 9
3 An example LAS with Gaussian approximation . . . 12
4 Discretization of the link descriptor . . . 16
5 A typical Wireless Sensor Network, with Base Station denoted as a gray box, multicast nodes as gray circles and regular nodes as white circles. . . . 19
6 The probability of longest route exceeding threshold T for two trees ♦ and . The approximated Chernoff-bound for these probabilities are ◊ and , respectively. . . . 24
7 An example for . . . 32
8 GEANT network topology used to evaluate the proposed algorithms. On the left side the Mercator projection on the right side a flattened representation of the topology can be seen. . . . 33
9 Example delay distributions on each link . . . 34
10 Path index vs its performance , T_{max}=3000. The left figure depicts all the paths, while the figure on the right zooms in to the first 20 best path. . . . 34
11 Path choice frequency out of 10^4 samples when the OSPF did not choose path #1 35
12 The distribution of the number of different paths between all *src* to all *dst* for the random graph ensemble . . . 35
13 The distributions of the ensemble performance metric G over all graphs G in the random graph ensemble for all for all introduced algorithms. . . . 36
14 The total ensemble performance metric $_e$ for all introduced algorithms. . . . 36
15 A typical evaluation of the $\tilde{A}_1$ objective function. . . . 37
16 Performance of the multicast tree finder algorithm in case of additive measures . 38
17 Example of the daily self similarity pattern in the real switch telemetry data from mlab1.dub01.measurement-lab.org. Moving averaged data was plotted from 2018. May 4-9. . . . 39
18 Packet rate and speed statistics aggregated over the time interval 14:00-15:00 each day in 2018. May. 1. to Jun. 30 and generated from the corresponding model 40
19 Packet length distribution aggregated over the time interval 14:00-15:00 each day in 2018. May. 1. to Jun. 30 . . . 40
20 Packet rate and speed statistics aggregated over the time interval 18:30-19:30 each day in 2018. Sep. 1. to Nov 30 and generated from the corresponding model 41
21 Packet length distribution aggregated over the time interval 18:30-19:30 each day in 2018. Sep. 1. to Nov 30 . . . 41
22 Information theoretic metrics for traffic situation 1 . . . 41
23 Information theoretic metrics for traffic situation 2 . . . 42
24 Search space of a 4 dimensional UBQP - vertices of a 4 dimensional hypercube . 46
25 Discarding the 2^{nd} dimensional from the 3 dimensional hypercube . . . 49
26 A hypergraph of a 4 dimensional UBQP . . . 49

DOI:10.15774/PPKE.ITK.2019.006

27 Flow graph representation of the algorithms . . . 50
28 Representations of a 3 dimensional problem . . . 51
29 Performance on the 5th problem from the ORLIB 50 dimensional problems. . . . 56
30 Performance on the 7th problem from the ORLIB 100 dimensional problems. . . . 56
31 Performance on the 2nd problem from the ORLIB 500 dimensional problems. . . . 57
32 Relative run times on the 2nd problem from the ORLIB 500 dimensional problems. 57
33 Performance on the 5th of the 50 dimensional problems. . . . 58
34 Performance on the 7th of the 100 dimensional problems. . . . 58
35 Performance on the 2nd of the 500 dimensional problems. . . . 59
36 Execution time comparison of the L01 algorithm with parallelization. . . . 60
37 Visualization of the scheduling problem . . . 62
38 TWT performance of algorithms versus heuristic parameter for a specific case J 10 . . . 63
39 Quadratic value performance of algorithms respect to the performance of HNN versus heuristic parameter for a specific case J 10 . . . 64
40 Relative average TWT from best solution in each iteration . . . 65
41 Transceiver system using spread codes for multiple access . . . 66
42 A typical channel response measured for chip time unit . . . 67
43 BER performance of the algorithms for 7 users . . . 68
44 Objective function performance of the algorithms for 7 users . . . 69
45 Flow graph representation of the optimal detector . . . 72
46 Flow graph representation of the optimal detector . . . 73
47 Equivalence of the FFNN with an encoding . . . 74
48 Flow graph representation of the detector using an arbitrary encoding . . . 74
49 Coding error of the interval halving method . . . 77
50 The architecture of the FFNN used in simulations . . . 78
51 Performance curves with parameter L 10 for four typical channels . . . 79
52 Performance curves with parameter L 14 for the four channel models . . . 80
53 SNR loss curves for the Bad Urban channel model . . . 81
54 Equivalence of the FFNN with an encoding . . . 88
55 Flow graph representation of the detector using an arbitrary encoding . . . 88
56 Block diagram of a DHNN . . . 90
57 The general architecture of an FFNN . . . 90
58 Limits for the log-moment generating function and existence of optimal s parameter 95
59 Sharpest Chernoff bounds for a Geometric distribution . . . 96
60 Sharpest Chernoff bounds for a Poisson distribution . . . 96
61 A Steiner tree of 4 points . . . 97
62 Example of two trees on the graph. . . . 98
63 An example for an incorrect and a correct CGSMT . . . 98

List of Tables

1 Problems investigated, used algorithmic tools and the related theses . . . 3

DOI:10.15774/PPKE.ITK.2019.006

2 A sequence of packet reception until both data and ACK are received. 20
3 Categorization of the algorithms 54
4 Performance comparison of BTS vs DA01 59
5 SNR loss for “Hilly Terrain” and “Rural Area” channels 81
6 SNR loss for “Typical Urban” and “Bad Urban” channels 81
3 Categorization of the algorithms 87
4 Performance analysis table for the ORLIB problems 102
5 Run time table for the ORLIB problems 103

List of Algorithms

1 Exhaustive-s algorithm 15
2 The Recursive Path Finder - *s* Finder Algorithm 18
3 Find optimal tree for end-to-end requirement 23
4 Pseudo code of the general UBQP solver algorithm 53
1 Exhaustive-s algorithm 85
2 The Recursive Path Finder - *s* Finder Algorithm 85
3 Find optimal tree for end-to-end requirement 86
4 Pseudo code of the general UBQP solver algorithm 87

References

[1] *802.3-2012*. 2012. DOI: 10.1109/ieeestd.2012.6419735. URL: http://dx.doi.org/10.1109/IEEESTD.2012.6419735.

[2] I.F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci. "Wireless sensor networks: a survey". In: *Computer Networks* 38.4 (2002), pp. 393 –422. ISSN: 1389-1286. DOI: http://dx.doi.org/10.1016/S1389-1286(01)00302-4. URL: http://www.sciencedirect.com/science/article/pii/S1389128601003024.

[3] G. Apostolopoulos, S. Kama, D. Williams, R. Guerin, A. Orda, and T. Przygienda. *QoS Routing Mechanisms and OSPF Extensions*. RFC 2676 (Experimental). 1999. URL: http://www.ietf.org/rfc/rfc2676.txt.

[4] O. Aumage, E. Brunet, N. Furmento, and R. Namyst. "NEW MADELEINE: a Fast Communication Scheduling Engine for High Performance Networks". In: *Parallel and Distributed Processing Symposium, 2007. IPDPS 2007. IEEE International*. 2007, pp. 1–8. DOI: 10.1109/IPDPS.2007.370476.

[5] G.A. Azim. "Neural Networks for Solving Quadratic Assignment Problems". In: *Neural Information Processing-Letters and Reviews* 10.3 (2006), pp. 27–34.

[6] KalaPraveen Bagadi and Susmita Das. "Neural network-based adaptive multiuser detection schemes in SDMA–OFDM system for wireless application". English. In: *Neural Computing and Applications* 23.3-4 (2013), pp. 1071–1082. ISSN: 0941-0643. DOI: 10.1007/s00521-012-1033-z. URL: http://dx.doi.org/10.1007/s00521-012-1033-z.

[7] F. Barahona. "A solvable case of quadratic 0-1 programming". In: *Discrete Applied Mathematics* 13.1 (1986), pp. 23–26.

[8] F. Barahona, M. Jünger, and G. Reinelt. "Experiments in quadratic 0–1 programming". In: *Mathematical Programming* 44.1 (1989), pp. 127–137.

[9] L.E. Baum, T. Petrie, G. Soules, and N. Weiss. "A maximization technique occuring in statistical Analyzis of Probabilistic function of Markov chains". In: *The Annals of Mathematical Statistics* 41.1 (1970), pp. 164–171. ISSN: 0003-4851.

[10] J. E. Beasley. "OR-Library: distributing test problems by electronic mail". In: *Journal of the Operational Research Society* 41.11 (1990), pp. 1069–1072.

[11] J.E. Beasley. "Heuristic algorithms for the unconstrained binary quadratic programming problem". In: *London, UK: Management School, Imperial College* (1998).

[12] Alain Billionnet and Sourour Elloumi. "Using a Mixed Integer Quadratic Programming Solver for the Unconstrained Quadratic 0-1 Problem". In: *Mathematical Programming* 109 (1 2007). 10.1007/s10107-005-0637-9, pp. 55–68. ISSN: 0025-5610. URL: http://dx.doi.org/10.1007/s10107-005-0637-9.

[13] Dario Bini, Beatrice Meini, S Steffé, and B Van Houdt. "Structured Markov chains solver: tool extension". In: (Jan. 2009). DOI: 10.4108/ICST.VALUETOOLS2009.7443.

[14] E. Boros, P. Hammer, and X. Sun. "The DDT method for quadratic 0-1 minimization". In: *RUTCOR Research Center, RRR* 39 (1989), p. 89.

[15] E. Boros, P.L. Hammer, and G. Tavares. “Local search heuristics for quadratic unconstrained binary optimization (QUBO)”. In: *Journal of Heuristics* 13.2 (2007), pp. 99–132.

[16] L. Cariou and J.-F. Helard. “MIMO Frequency Hopping Spread Spectrum Multi-Carrier Multiple Access: A Novel Uplink System for B3G Cellular Networks”. English. In: *Telecommunication Systems* 30.1-3 (2005), pp. 193–214. ISSN: 1018-4864. DOI: 10.1007/s11235-005-4325-0. URL: http://dx.doi.org/10.1007/s11235-005-4325-0.

[17] Miguel A Carreira-Perpinan and Geoffrey E Hinton. “On Contrastive Divergence Learning.” In: *AISTATS*. Vol. 10. Citeseer. 2005, pp. 33–40.

[18] Shigang Chen and Klara Nahrstedt. “An overview of quality of service routing for next-generation high-speed networks: problems and solutions”. In: *IEEE Network Magazine, Special Issue on Transmission and Distribution of Digital Video* 12.6 (Nov. 1998), pp. 64–79. ISSN: 0890-8044. DOI: 10.1109/65.752646.

[19] Shigang Chen and Klara Nahrstedt. “On finding multi-constrained paths”. In: *Communications, 1998. ICC 98. Conference Record. 1998 IEEE International Conference on*. Vol. 2. IEEE. June 1998, pp. 874–879. DOI: 10.1109/ICC.1998.685137. URL: http://citeseer.nj.nec.com/chen98finding.html.

[20] Xiuzhen Cheng and Ding-Zhu Du. *Steiner Trees in Industry (Combinatorial Optimization)*. Springer, 2002. ISBN: 1402000995.

[21] Francisco Chicano and Enrique Alba. “Elementary landscape decomposition of the 0-1 unconstrained quadratic optimization”. English. In: *Journal of Heuristics* (2011), pp. 1–18. ISSN: 1381-1231. DOI: 10.1007/s10732-011-9170-6. URL: http://dx.doi.org/10.1007/s10732-011-9170-6.

[22] Young Hwan Lee; Chae Hun Chung; Sung Ho Cho. “Performance analysis of a convolutional coded DS/CDMA system in Nakagami fading channels”. English. In: *Telecommunication Systems* 14.1-4 (1-4 2000), pp. 31–45. ISSN: 1018-4864. DOI: 10.1023/A:1019125029959. URL: http://dx.doi.org/10.1023/A%3A1019125029959.

[23] Hongsik Choi, Ju Wang, and EstherA. Hughes. “Scheduling for information gathering on sensor network”. English. In: *Wireless Networks* 15 (1 2009), pp. 127–140. ISSN: 1022-0038. DOI: 10.1007/s11276-007-0050-9. URL: http://dx.doi.org/10.1007/s11276-007-0050-9.

[24] Richard Courant and H. Robbins. *What Is Mathematics? : An Elementary Approach to Ideas and Methods*. Oxford University Press, 1996. ISBN: 0195105192.

[25] G. Cybenko. “Approximation by superpositions of a sigmoidal function”. English. In: *Mathematics of Control, Signals and Systems* 2.4 (1989), pp. 303–314. ISSN: 0932-4194. DOI: 10.1007/BF02551274. URL: http://dx.doi.org/10.1007/BF02551274.

[26] George Dahl, Dong Yu, Li Deng, and Alex Acero. “Context-Dependent Pre-trained Deep Neural Networks for Large Vocabulary Speech Recognition”. In: vol. 20. 2012, pp. 30–42.

[27] M.O. Damen, H. El Gamal, and G. Caire. “On maximum-likelihood detection and the search for the closest lattice point”. In: *Information Theory, IEEE Transactions on* 49.10 (2003), pp. 2389 –2402. ISSN: 0018-9448. DOI: 10.1109/TIT.2003.817444.

[28] C. De Simone. “The cut polytope and the boolean quadric polytope”. In: *Discrete Mathematics* 79.1 (1990), pp. 71–75.

[29] *deep dream codebase on github*. https://github.com/google/deepdream. Accessed: 2017.02.24.

[30] *deep dream generator*. https://deepdreamgenerator.com. Accessed: 2017.02.24.

[31] *Deep Learning Tutorial - LISA lab, University of Montreal*. http://deeplearning.net/tutorial/deeplearning.pdf. Accessed: 2017.02.24.

[32] Li Deng and Dong Yu. *Deep Learning: Methods and Applications*. Tech. rep. 2014.

[33] T Dogan, J Zheng, A Engelhart, MA Dangl, and H Schummer. *Visualization of the discrete-time channel matrix (including an animation of the transmission)*. 2002. URL: http://it.e-technik.uni-ulm.de/Rdemo/index.html.

[34] Achim Engelhart, Werner G Teich, Jürgen Lindner, Gábor Jeney, Sándor Imre, and László Pap. “A survey of multiuser/multisubchannel detection schemes based on recurrent neural networks”. In: *Wireless Communications and Mobile Computing* 2.3 (2002), pp. 269–284.

[35] M. Failli. *Digital land mobile radio communications COST 207*. Tech. rep. European Commission, 1989.

[36] Shen Fang, Sun Yunshan, and Zhang Liyi. “A Novel Neural Network Blind Multi-user Detection Algorithm”. In: *Industrial Electronics and Applications, 2007. ICIEA 2007. 2nd IEEE Conference on*. 2007, pp. 1547–1550. DOI: 10.1109/ICIEA.2007.4318667.

[37] Wolfgang Fischer and Kathleen Meier-Hellstern. “The Markov-modulated Poisson Process (MMPP) Cookbook”. In: *Perform. Eval.* 18.2 (Sept. 1993), pp. 149–171. ISSN: 0166-5316. DOI: 10.1016/0166-5316(93)90035-S. URL: https://doi.org/10.1016/0166-5316(93)90035-S.

[38] Norbert Fogarasi, Kalman Tornai, and Janos Levendovszky. “A Novel Hopfield Neural Network Approach for Minimizing Total Weighted Tardiness of Jobs Scheduled on Identical Machines”. In: *Acta Univ. Sapientiae Informatica* 4.1 (2012), pp. 48–66.

[39] N. Funabiki and S. Nishikawa. “A binary Hopfield neural-network approach for satellite broadcast scheduling problems”. In: *Neural Networks, IEEE Transactions on* 8.2 (1997), pp. 441 –445. ISSN: 1045-9227. DOI: 10.1109/72.557699.

[40] N. Funabiki, Y. Takefuji, and K.C. Lee. “Comparisons of seven neural network models on traffic control problems in multistage interconnection networks”. In: *Computers, IEEE Transactions on* 42.4 (1993), pp. 497 –501. ISSN: 0018-9340. DOI: 10.1109/12.214695.

[41] Nobuo Funabiki and Junji Kitamichi. “A gradual neural network algorithm for broadcast scheduling problems in packet radio networks”. In: *IEICE Transactions on Fundamentals* (1999), pp. 815–824.

[42] Ken-Ichi Funahashi. “On the approximate realization of continuous mappings by neural networks”. In: *Neural Networks* 2.3 (1989), pp. 183 –192. ISSN: 0893-6080. DOI: http://dx.doi.org/10.1016/0893-6080(89)90003-8. URL: http://www.sciencedirect.com/science/article/pii/0893608089900038.

[43] H. R. Gail, S. L. Hantler, and B. A. Taylor. “Non-Skip-Free M/G/1 and G/M/1 Type Markov Chains”. In: *Advances in Applied Probability* 29.3 (1997), pp. 733–758. ISSN: 00018678. URL: http://www.jstor.org/stable/1428084.

[44] Azim Gamil A. “Neural Networks for Solving Quadratic Assignment Problems”. In: *Neural Information Processing - Letters and Reviews* 10.3 (2006), pp. 51 –57. ISSN: 1738-2572.

[45] Michael R. Garey and David S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. Vol. 174. New York, NY, USA: W.H.Freeman & Co Ltd, 1979. ISBN: 0716710447.

[46] G.Casale, E.Z.Zhang, and E.Smirni. “KPC-Toolbox: Best Recipes for Automatic Trace Fitting Using Markovian Arrival Processes”. In: vol. 67. Sept. 2010, 873–896 vol 9.

[47] G.Casale, E.Z.Zhang, and E.Smirni. “Trace Data Characterization and Fitting for Markov Modeling”. In: vol. 67. Feb. 2010, 61–79 vol 2.

[48] *GEANT european network topology*. Accessed: 2019.01.19. URL: https://www.geant.org/Resources/Documents/G%C3%89ANT_Topology_Map_January_2018.pdf.

[49] *Geoffrey E. Hinton’s Publications in Reverse Chronological Order*. http://www.cs.toronto.edu/~hinton/papers.html. Accessed: 2017.02.24.

[50] Wulfram Gerstner, Werner M. Kistler, Richard Naud, and Liam Paninski. *Neuronal Dynamics: From Single Neurons to Networks and Models of Cognition*. Cambridge University Press, 2014. ISBN: 9781107447615. DOI: 10.1017/CBO9781107447615. URL: http://neuronaldynamics.epfl.ch/online/.

[51] Patrick O. Glauner. “Deep Convolutional Neural Networks for Smile Recognition”. In: *CoRR* abs/1508.06535 (2015). URL: http://arxiv.org/abs/1508.06535.

[52] Xavier Glorot and Yoshua Bengio. “Understanding the difficulty of training deep feedforward neural networks”. In: *In Proceedings of the International Conference on Artificial Intelligence and Statistics (AISTATS’10). Society for Artificial Intelligence and Statistics*. 2010.

[53] F. Glover, G.A. Kochenberger, and B. Alidaee. “Adaptive memory tabu search for binary quadratic programs”. In: *Management Science* 44 (1998), pp. 336–345.

[54] F. Glover, B. Alidaee, C. Rego, and G. Kochenberger. “One-pass heuristics for large-scale unconstrained binary quadratic problems”. In: *European Journal of Operational Research* 137.2 (2002), pp. 272–287.

[55] M. Goguen. “Private Network-Network Interface Specification Version 1.0”. In: *PNNI Specification Working Group ATM forum, March*. 1996.

[56] R. Gold. “Optimal binary sequences for spread spectrum multiplexing (Corresp.)” In: *Information Theory, IEEE Transactions on* 13.4 (1967), pp. 619 –621. ISSN: 0018-9448. DOI: 10.1109/TIT.1967.1054048.

[57] Lee Gomes. *Machine-Learning Maestro Michael Jordan on the Delusions of Big Data and Other Huge Engineering Efforts*. http://spectrum.ieee.org/robotics/artificial-intelligence/machinelearning-maestro-michael-jordan-on-the-delusions-of-big-data-and-other-huge-engineering-efforts. Accessed: 2017.02.24. 2014.

[58] Roche A Guérin and Ariel Orda. “QoS routing in networks with inaccurate information: theory and algorithms”. In: *IEEE/ACM Transactions on Networking (TON)* 7.3 (1999), pp. 350–364.

[59] Roch Guérin and Ariel Orda. “Computing shortest paths for any number of hops”. In: *Networking, IEEE/ACM Transactions on* 10.5 (2002), pp. 613–620. ISSN: 1063-6692. DOI: http://dx.doi.org/10.1109/TNET.2002.803917.

[60] M. Haenggi. “On Routing in Random Rayleigh Fading Networks”. In: *IEEE Transactions on Wireless Communications* 4.4 (2005). Available at http://www.nd.edu/~mhaenggi/pubs/routing.pdf., pp. 1553–1562.

[61] Saïd Hanafi, Ahmed-Riadh Rebai, and Michel Vasquez. “Several versions of the devour digest tidy-up heuristic for unconstrained binary quadratic problems”. English. In: *Journal of Heuristics* (2011), pp. 1–33. ISSN: 1381-1231. DOI: 10.1007/s10732-011-9169-z. URL: http://dx.doi.org/10.1007/s10732-011-9169-z.

[62] Simon Haykin. *Communication Systems*. Wiley, 2009. ISBN: 0471697907.

[63] Simon Haykin. *Neural networks: a comprehensive foundation*. Prentice Hall PTR, 1994. ISBN: 0132733501.

[64] Simon Haykin. *Neural networks and learning machines*. Prentice Hall PTR, 2009. ISBN: 0131471392.

[65] Simon Haykin and Michael Moher. *An Introduction to Analog and Digital Communications*. Wiley, 2006. ISBN: 0471432229.

[66] Li He. “A Neural Network Blind Multi-user Detection Algorithm”. In: *Measuring Technology and Mechatronics Automation, 2009. ICMTMA ’09. International Conference on*. Vol. 2. 2009, pp. 467–470. DOI: 10.1109/ICMTMA.2009.92.

[67] H. Heffes and D. Lucantoni. “A Markov Modulated Characterization of Packetized Voice and Data Traffic and Related Statistical Multiplexer Performance”. In: *IEEE Journal on Selected Areas in Communications* 4.6 (1986), pp. 856–868. ISSN: 0733-8716. DOI: 10.1109/JSAC.1986.1146393.

[68] C. Helmberg and F. Rendl. “Solving quadratic (0, 1)-problems by semidefinite programs and cutting planes”. In: *Mathematical Programming* 82.3 (1998), pp. 291–315.

[69] Geoffrey E Hinton. *A Practical Guide to Training Restricted Boltzmann Machines*. http://www.cs.toronto.edu/~hinton/absps/guideTR.pdf. Accessed: 2017.02.24. 2010.

[70] Geoffrey E Hinton. *Recent Developments in Deep Learning - Google Tech Talks*. https://www.youtube.com/watch?v=VdIURAu1-aU. Accessed: 2017.02.24. 2010.

[71] Geoffrey E Hinton. *The Next Generation of Neural Networks - Google Tech Talks*. https://www.youtube.com/watch?v=AyzOUbkUf3M. Accessed: 2017.02.24. 2007.

[72] Geoffrey E Hinton, Simon Osindero, and Yee-Whye Teh. “A fast learning algorithm for deep belief nets”. In: *Neural computation* 18.7 (2006), pp. 1527–1554.

[73] Geoffrey E Hinton and Ruslan R Salakhutdinov. “Reducing the dimensionality of data with neural networks”. In: *science* 313.5786 (2006), pp. 504–507.

[74] *homepage of Prof. dr. Benny Van Houdt*. https://win.uantwerpen.be/~vanhoudt/. Accessed: 2019.05.14.

[75] J.J. Hopfield and D.W. Tank. “Neural computation of decisions in optimization problems”. In: *Biological Cybernetics* 52.3 (3 1985), pp. 141–152. ISSN: 0340-1200.

[76] Kurt Hornik, Maxwell Stinchcombe, and Halbert White. “Multilayer feedforward networks are universal approximators”. In: *Neural networks* 2.5 (1989), pp. 359–366.

[77] *How Deep Learning Can Paint Videos in the Style of Art’s Great Masters*. https://blogs.nvidia.com/blog/2016/05/25/deep-learning-paints-videos/. Accessed: 2017.02.24. 2016.

[78] J.Y. Hui. “Resource allocation for broadband networks”. In: *Selected Areas in Communications, IEEE Journal on* 6.9 (Dec. 1988), pp. 1598 –1608. ISSN: 0733-8716. DOI: 10.1109/49.12887.

[79] Ho-Lung Hung. “Interference cancellation for HNNPSO multiuser detection of UWB systems over multipath fading channel”. English. In: *Telecommunication Systems* 52.2 (2013), pp. 1191–1203. ISSN: 1018-4864. DOI: 10.1007/s11235-011-9634-x. URL: http://dx.doi.org/10.1007/s11235-011-9634-x.

[80] Ho-Lung Hung and Yung-Fa Huang. “Performance of multiuser detectors based on EM-like method for ultra-wideband communications systems in multipath fading channel”. English. In: *Telecommunication Systems* (2013), pp. 1–14. ISSN: 1018-4864. DOI: 10.1007/s11235-013-9694-1. URL: http://dx.doi.org/10.1007/s11235-013-9694-1.

[81] F. K. Hwang and Dana S. Richards. “Steiner tree problems”. In: *Networks* 22.1 (1992), pp. 55–89. ISSN: 1097-0037. DOI: 10.1002/net.3230220105. URL: http://dx.doi.org/10.1002/net.3230220105.

[82] *IEEE Standard for Information technology–Telecommunications and information exchange between systems Local and metropolitan area networks–Specific requirements Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications*. IEEE -Org, 2012. DOI: 10.1109/ieeestd.2012.6178212. URL: http://dx.doi.org/10.1109/IEEESTD.2012.6178212.

[83] S.M.Riazul Islam, M.A. Ameen, and KyungSup Kwak. “Channel estimation in ECMA-368-based UWB systems with unknown interference”. English. In: *Telecommunication Systems* 52.2 (2013), pp. 1159–1169. ISSN: 1018-4864. DOI: 10.1007/s11235-011-9631-0. URL: http://dx.doi.org/10.1007/s11235-011-9631-0.

[84] *ISO/IEC standard 7498-1:1994 - OSI/ISO 7 layer model*. http://standards.iso.org/ittf/PubliclyAvailableStandards/s020269_ISO_IEC_7498-1_1994(E).zip.

[85] J.M. Jaffe. “Algorithms for finding paths with multiple constraints”. In: *Networks* 14.1 (1984), pp. 95–116.

[86] J. Jalden and B. Ottersten. “On the complexity of sphere decoding in digital communications”. In: *Signal Processing, IEEE Transactions on* 53.4 (2005), pp. 1474–1484. ISSN: 1053-587X. DOI: 10.1109/TSP.2005.843746.

[87] Luo Junhai, Xue Liu, and Ye Danxia. “Research on multicast routing protocols for mobile ad-hoc networks”. In: *Computer Networks* 52.5 (2008), pp. 988 –997. ISSN: 1389-1286. DOI: DOI:10.1016/j.comnet.2007.11.016.

[88] Salim Kahveci. “A channel detection technique based on Max-Log-MAP algorithm for wireless indoor system”. English. In: *Telecommunication Systems* 49.4 (2012), pp. 345–

353. ISSN: 1018-4864. DOI: 10.1007/s11235-010-9382-3. URL: http://dx.doi.org/10.1007/s11235-010-9382-3.

[89] Mirosław Kantor, Piotr Chołda, and Andrzej Jajszczyk. "Least Cost Routing (LCR) solution for inter-domain traffic distribution". In: *Telecommunication Systems* (2011). 10.1007/s11235-011-9606-1, pp. 1–13. ISSN: 1018-4864. URL: http://dx.doi.org/10.1007/s11235-011-9606-1.

[90] John Klincewicz, James Schmitt, and Richard Wong. "Incorporating QoS into IP Enterprise Network Design". In: *Telecommunication Systems* 20 (1 2002). 10.1023/A:1015441400785, pp. 81–106. ISSN: 1018-4864. URL: http://dx.doi.org/10.1023/A:1015441400785.

[91] G.A. Kochenberger, F. Glover, B. Alidaee, and C. Rego. "A unified modeling and solution framework for combinatorial optimization problems". In: *OR Spectrum* 26.2 (2004), pp. 237–250.

[92] GaryA. Kochenberger, Fred Glover, Bahram Alidaee, and Cesar Rego. "Solving Combinatorial Optimization Problems Via Reformulation and Adaptive Memory Metaheuristics". In: *Frontiers of Evolutionary Computation*. Ed. by Anil Menon. Vol. 11. Genetic Algorithms and Evolutionary Computation. Springer US, 2004, pp. 103–113. ISBN: 978-1-4020-7524-7. DOI: 10.1007/1-4020-7782-3_5. URL: http://dx.doi.org/10.1007/1-4020-7782-3_5.

[93] Vachaspathi P. Kompella, Joseph C. Pasquale, and George C. Polyzos. "Multicast routing for multimedia communication". In: *IEEE/ACM Trans. Netw.* 1.3 (1993), pp. 286–292. ISSN: 1063-6692. DOI: http://dx.doi.org/10.1109/90.234851.

[94] Vachaspathi Peter Kompella. "Multicast routing algorithms for multimedia traffic". PhD thesis. La Jolla, CA, USA: University of California at San Diego, 1993.

[95] *KPC-toolbox*. http://www.cs.wm.edu/MAPQN/kpctoolbox.html. Accessed: 2019.05.14.

[96] E.G. Larsson. "MIMO Detection Methods: How They Work [Lecture Notes]". In: *Signal Processing Magazine, IEEE* 26.3 (2009), pp. 91 –95. ISSN: 1053-5888. DOI: 10.1109/MSP.2009.932126.

[97] G. Latouche and V. Ramaswami. *Introduction to Matrix Analytic Methods in Stochastic Modeling*. Society for Industrial and Applied Mathematics, 1999. DOI: 10.1137/1.9780898719734. eprint: https://epubs.siam.org/doi/pdf/10.1137/1.9780898719734. URL: https://epubs.siam.org/doi/abs/10.1137/1.9780898719734.

[98] Yann LeCun, Leon Bottou, Genevieve B. Orr, and Klaus Robert Müller. "Efficient BackProp". In: *Neural Networks: Tricks of the Trade*. Ed. by Genevieve B. Orr and Klaus-Robert Müller. Berlin, Heidelberg: Springer Berlin Heidelberg, 1998, pp. 9–50. ISBN: 978-3-540-49430-0. DOI: 10.1007/3-540-49430-8_2. URL: http://dx.doi.org/10.1007/3-540-49430-8_2.

[99] Whay Chiou Lee. "Spanning tree method for link state aggregation in large communication networks". In: *INFOCOM'95. Fourteenth Annual Joint Conference of the IEEE Computer and Communications Societies. Bringing Information to People. Proceedings. IEEE*. IEEE.

Washington, DC, USA: IEEE Computer Society, 1995, pp. 297–302. ISBN: 0-8186-6990-X.

[100] J Levendovszky and EC van der Meulen. “Nonparametric bayesian estimation by feedforward neural networks”. In: *Prague conference on information theory, statistical decision functions and random processes*. 1998.

[101] J. Levendovszky, K. Tornai, G. Treplan, and A. Olah. “Novel Load Balancing Algorithms Ensuring Uniform Packet Loss Probabilities for WSN”. In: *Vehicular Technology Conference (VTC Spring), 2011 IEEE 73rd*. 2011, pp. 1–5. DOI: 10.1109/VETECS.2011.5956703.

[102] Janos Levendovszky and Edward C. van der Meulen. “Tail Distribution Estimation for Call Admission Control in ATM Networks”. In: *Proccedings of IFIP, Third Workshop on Performance Modelling and Evaluation of ATM Networks*. July 1995.

[103] János Levendovszky, Alpár Fancsali, Csaba Végso, and Gábor Rétvári. “QoS Routing with Incomplete Information by Analog Computing Algorithms”. In: *Quality of Future Internet Services: Second COST 263 International Workshop, QofIS 2001 Coimbra, Portugal, September 24–26, 2001 Proceedings*. Ed. by Mikhail I. Smirnov, Jon Crowcroft, James Roberts, and Fernando Boavida. Berlin, Heidelberg: Springer Berlin Heidelberg, 2001, pp. 127–137. ISBN: 978-3-540-45412-0. DOI: 10.1007/3-540-45412-8_10. URL: http://dx.doi.org/10.1007/3-540-45412-8_10.

[104] Huan Li, P. Shenoy, and K. Ramamritham. “Scheduling messages with deadlines in multi-hop real-time sensor networks”. In: *Real Time and Embedded Technology and Applications Symposium, 2005. RTAS 2005. 11th IEEE*. 2005, pp. 415–425. DOI: 10.1109/RTAS.2005.48.

[105] Wen Liu and Lipo Wang. “Solving the Shortest Path Routing Problem Using Noisy Hopfield Neural Networks”. In: *Communications and Mobile Computing, 2009. CMC ’09. WRI International Conference on*. Vol. 2. Jan. 2009, pp. 299 –302. DOI: 10.1109/CMC.2009.366.

[106] A. Lodi, K. Allemand, and T.M. Liebling. “An evolutionary heuristic for quadratic 0-1 programming”. In: *European Journal of Operational Research* 119.3 (1999), pp. 662–670.

[107] Dean H. Lorenz and Ariel Orda. “QoS routing in networks with uncertain parameters”. In: *IEEE/ACM Transactions on Networking* 6.6 (Dec. 1998), pp. 768–778. ISSN: 1063-6692. DOI: 10.1109/90.748088. URL: citeseer.ist.psu.edu/lorenz98qos.html.

[108] Z. Luo, W. Ma, A.M.C. So, Y. Ye, and S. Zhang. “Semidefinite relaxation of quadratic optimization problems”. In: *Signal Processing Magazine, IEEE* 27.3 (2010), pp. 20–34.

[109] Nguyen Cong Luong, Dinh Thai Hoang, Ping Wang, Dusit Niyato, Dong In Kim, and Zhu Han. “Data collection and wireless communication in Internet of Things (IoT) using economic analysis and pricing models: A survey”. In: *IEEE Communications Surveys & Tutorials* 18.4 (2016), pp. 2546–2590.

[110] *MAMSolver*. http://www.cs.wm.edu/MAMSolver/. Accessed: 2019.05.14.

[111] Haci Mantar. “A scalable QoS routing model for diffserv over MPLS networks”. In: *Telecommunication Systems* 34 (3 2007). 10.1007/s11235-007-9035-3, pp. 107–115. ISSN: 1018-4864. URL: http://dx.doi.org/10.1007/s11235-007-9035-3.

[112] Stéphane Martignoni and Thomas Kühnel. “Extension of Classical IP over ATM to support QoS at the application level”. In: *Telecommunication Systems* 11 (3 1999). 10.1023/A:1019161721356, pp. 291–303. ISSN: 1018-4864. URL: http://dx.doi.org/10.1023/A:1019161721356.

[113] Warren S. McCulloch and Walter Pitts. “A logical calculus of the ideas immanent in nervous activity”. In: *The bulletin of mathematical biophysics* 5.4 (1943), pp. 115–133. ISSN: 1522-9602. DOI: 10.1007/BF02478259. URL: http://dx.doi.org/10.1007/BF02478259.

[114] *MeasurementLab disco dataset blogpost*. https://www.measurementlab.net/blog/disco-dataset/#new-disco-switch-telemetry-dataset. Accessed: 2019.05.14.

[115] *MeasurementLab homepage*. https://www.measurementlab.net. Accessed: 2019.05.14.

[116] P. Merz and B. Freisleben. “Genetic algorithms for binary quadratic programming”. In: *Proceedings of the genetic and evolutionary computation conference*. Vol. 1. Citeseer. 1999, pp. 417–424.

[117] Peter Merz and Bernd Freisleben. “Greedy and Local Search Heuristics for Unconstrained Binary Quadratic Programming”. In: *Journal of Heuristics* 8 (2 2002). 10.1023/A:1017912624016, pp. 197–213. ISSN: 1381-1231. URL: http://dx.doi.org/10.1023/A:1017912624016.

[118] Peter Merz and Kengo Katayama. “Memetic algorithms for the unconstrained binary quadratic programming problem”. In: *Biosystems* 78.1–3 (2004), pp. 99 –118. ISSN: 0303-2647. DOI: 10.1016/j.biosystems.2004.08.002. URL: http://www.sciencedirect.com/science/article/pii/S0303264704001376.

[119] H.N. Mhaskar. “Approximation properties of a multilayered feedforward artificial neural network”. English. In: *Advances in Computational Mathematics* 1.1 (1993), pp. 61–80. ISSN: 1019-7168. DOI: 10.1007/BF02070821. URL: http://dx.doi.org/10.1007/BF02070821.

[120] Luca Muscariello, M Meillia, M Meo, M Ajmone Marsan, and Renato Lo Cigno. “An MMPP-based hierarchical model of Internet traffic”. In: vol. 4. July 2004, 2143 –2147 Vol.4. ISBN: 0-7803-8533-0. DOI: 10.1109/ICC.2004.1312897.

[121] Hou Muzhou, Han Xuli, and Gan Yixuan. “Constructive approximation to real function by wavelet neural networks”. English. In: *Neural Computing and Applications* 18.8 (2009), pp. 883–889. ISSN: 0941-0643. DOI: 10.1007/s00521-008-0194-2. URL: http://dx.doi.org/10.1007/s00521-008-0194-2.

[122] M.F. Neuts. *Matrix-geometric Solutions in Stochastic Models: An Algorithmic Approach*. Algorithmic Approach. Dover Publications, 1994. ISBN: 9780486683423.

[123] Takayuki Osogami. “Analysis of Multi-server Systems via Dimensionality Reduction of Markov Chains”. AAI3171954. PhD thesis. Pittsburgh, PA, USA, 2005. ISBN: 0-542-08412-0. URL: http://www.cs.cmu.edu/~osogami/thesis/.

[124] Toshihisa Ozawa. “Sojourn time distributions in the queue defined by a general QBD process”. In: *Queueing Systems* 53.4 (2006), pp. 203–211. ISSN: 1572-9443. DOI: 10.

1007/s11134-006-7651-3. URL: https://doi.org/10.1007/s11134-006-7651-3.

[125] Battal Özdemir and Özgür Gürbüz. "Tomlinson-Harashima Precoded MIMO in wireless networks: to THP or not to THP?" English. In: *Telecommunication Systems* (2013), pp. 1–13. ISSN: 1018-4864. DOI: 10.1007/s11235-013-9705-2. URL: http://dx.doi.org/10.1007/s11235-013-9705-2.

[126] G. Palubeckis. "Iterated tabu search for the unconstrained binary quadratic optimization problem". In: *Informatica* 17.2 (2006), pp. 279–296.

[127] J.C. Picard and H.D. Ratliff. "A graph-theoretic equivalence for integer programs". In: *Operations Research* (1973), pp. 261–269.

[128] J.C. Picard and H.D. Ratliff. "Minimum cuts and related problems". In: *Networks* 5.4 (1975), pp. 357–370.

[129] S. Poljak, F. Rendl, and H. Wolkowicz. "A recipe for semidefinite relaxation for (0, 1)-quadratic programming". In: *Journal of Global Optimization* 7.1 (1995), pp. 51–73.

[130] C. Pornavalai, G. Chakraborty, and N. Shiratori. "A neural network approach to multicast routing in real-time communication networks". In: *ICNP'95: Proceedings of the 1995 International Conference on Network Protocols*. IEEE Computer Society, 1995, p. 332. ISBN: 0-8186-7216-1.

[131] Chotipat Pornavalai, Goutam Chakraborty, and Norio Shiratori. "Routing with multiple QoS requirements for supporting multimedia applications". In: *Telecommunication Systems* 9 (3 1998). 10.1023/A:1019160226383, pp. 357–373. ISSN: 1018-4864. URL: http://dx.doi.org/10.1023/A:1019160226383.

[132] John Proakis and Masoud Salehi. *Digital Communications, 5th Edition*. McGraw-Hill Science/Engineering/Math, 2007. ISBN: 0072957166.

[133] J. F. Pérez, J. Van Velthoven, and B. Van Houdt. *Q-MAM: a tool for solving infinite queues using matrix-analytic methods*. 2008.

[134] *Q-MAM solver*. https://win.uantwerpen.be/~vanhoudt/tools/QMAM.zip. Accessed: 2019.05.14.

[135] L.R. Rabiner. "A tutorial on hidden Markov models and selected applications in speech recognition". In: *Proceedings of the IEEE* 77.2 (Feb. 1989), pp. 257 –286. ISSN: 0018-9219. DOI: 10.1109/5.18626.

[136] Bhaskar Prasad Rimal, Eunmi Choi, and Ian Lumb. "A taxonomy and survey of cloud computing systems". In: *INC, IMS and IDC, 2009. NCM'09. Fifth International Joint Conference on*. Ieee. 2009, pp. 44–51.

[137] Alma Riska and Evgenia Smirni. "ETAQA Solutions for Infinite Markov Processes with Repetitive Structure". In: *INFORMS JOURNAL ON COMPUTING* 19.2 (2007), pp. 215–228. DOI: 10.1287/ijoc.1050.0160. eprint: http://joc.journal.informs.org/cgi/reprint/19/2/215.pdf. URL: http://joc.journal.informs.org/cgi/content/abstract/19/2/215.

[138] G. Robins and Zelikovsky. "A. Minimum Steiner Tree Construction". In: *The Handbook of Algorithms for VLSI Physical Design Automation* (2009). Ed. by D. P. Mehta C. J. Alpert and S. S. Sapatnekar, pp. 487–508.

[139] Maria Alejandra Rodriguez and Rajkumar Buyya. “A taxonomy and survey on scheduling algorithms for scientific workflows in IaaS cloud computing environments”. In: *Concurrency and Computation: Practice and Experience* (2016).

[140] M. Ruder, A. Dosovitskiy, and T. Brox. “Artistic style transfer for videos”. In: *ArXiv e-prints* (Apr. 2016). arXiv: 1604.08610 [cs.CV].

[141] T. Ryden. “Parameter estimation for Markov modulated Poisson processes”. In: *Stochastic Models* 10.4 (1994), pp. 795–829. ISSN: 1532-6349.

[142] Claudio Sacchi and Massimiliano Panizza. “Multi-rate group-orthogonal OFDMA-CDMA for broadband mobile transmission”. English. In: *Telecommunication Systems* 52.1 (2013), pp. 15–29. ISSN: 1018-4864. DOI: 10.1007/s11235-011-9441-4. URL: http://dx.doi.org/10.1007/s11235-011-9441-4.

[143] Ruslan Salakhutdinov, Andriy Mnih, and Geoffrey Hinton. “Restricted Boltzmann Machines for Collaborative Filtering”. In: *Proceedings of the 24th International Conference on Machine Learning*. ICML ’07. Corvalis, Oregon, USA: ACM, 2007, pp. 791–798. ISBN: 978-1-59593-793-3. DOI: 10.1145/1273496.1273596. URL: \url{http://doi.acm.org/10.1145/1273496.1273596}.

[144] Hussein F. Salama, Douglas S. Reeves, and Yannis Viniotis. “Evaluation of multicast routing algorithms for real-time communication on high-speed networks”. In: *IEEE Journal on Selected Areas in Communications* 15 (1997), pp. 332–345.

[145] A. Shaikh, J. Rexford, and K. Shin. “Efficient precomputation of quality-of-service routes”. In: *Proc. Workshop on Network and Operating Systems Support for Digital Audio and Video*. 1998, pp. 15–27. URL: citeseer.ist.psu.edu/article/shaikh98efficient.html.

[146] Anees Shaikh, Jennifer Rexford, and Kang G Shin. “Dynamics of quality-of-service routing with inaccurate link-state information”. In: *Ann Arbor* 1001.CSE-TR-350-97 (1997), pp. 48109–2122. URL: citeseer.ist.psu.edu/shaikh97dynamics.html.

[147] Sukhpal Singh and Inderveer Chana. “A survey on resource scheduling in cloud computing: Issues and challenges”. In: *Journal of Grid Computing* 14.2 (2016), pp. 217–264.

[148] *SMC Solver MG1*. https://win.uantwerpen.be/~vanhoudt/tools/MG1files.zip. Accessed: 2019.05.14.

[149] *SMC Solver QBD*. https://win.uantwerpen.be/~vanhoudt/tools/QBDfiles.zip. Accessed: 2019.05.14.

[150] K. Smith, M. Palaniswami, and M. Krishnamoorthy. “Neural techniques for combinatorial optimization with applications”. In: *Neural Networks, IEEE Transactions on* 9.6 (1998), pp. 1301–1318.

[151] Kate A. Smith. “Neural Networks for Combinatorial Optimization: A Review of More Than a Decade of Research”. In: *INFORMS Journal on Computing* 11.1 (1999), pp. 15–34. DOI: 10.1287/ijoc.11.1.15. eprint: http://joc.journal.informs.org/content/11/1/15.full.pdf+html. URL: http://dx.doi.org/10.1287/ijoc.11.1.15.

[152] Paul Smolensky. “Information Processing in Dynamical Systems: Foundations of Harmony Theory”. In: *Parallel distributed processing: Explorations in the microstructure of cognition*. MIT Press, 1986, pp. 194–281. ISBN: 0-262-68053-X.

[153] Larry R. Squire, Darwin Berg, Floyd E. Bloom, Sascha du Lac, Anirvan Ghosh, and Nicholas C. Spitzer, eds. *Fundamental Neuroscience (Fourth Edition)*. Fourth Edition. San Diego: Academic Press, 2013, pp. i –ii. ISBN: 978-0-12-385870-2. DOI: http://dx.doi.org/10.1016/B978-0-12-385870-2.00054-8.

[154] *Structured Stochastic Matrices of M/G/1 Type and Their Applications*. Probability: Pure and Applied. Taylor & Francis, 1989. ISBN: 9780824782832.

[155] Shin Suzuki. “Function Approximation by Three-Layer Artificial Neural Networks”. English. In: *Mathematics of Neural Networks*. Ed. by StephenW. Ellacott, JohnC. Mason, and IainJ. Anderson. Vol. 8. Operations Research/Computer Science Interfaces Series. Springer US, 1997, pp. 349–354. ISBN: 978-1-4613-7794-8. DOI: 10.1007/978-1-4615-6099-9_61. URL: http://dx.doi.org/10.1007/978-1-4615-6099-9_61.

[156] Christian Szegedy et al. “Going Deeper with Convolutions”. In: *CoRR* abs/1409.4842 (2014). URL: http://arxiv.org/abs/1409.4842.

[157] Tim Szigeti and Christina Hattingh. *End-to-end qos network design*. Cisco press, 2005.

[158] D. Tank and J. Hopfield. “Simple ’neural’ optimization networks: An A/D converter, signal decision circuit, and a linear programming circuit”. In: *Circuits and Systems, IEEE Transactions on* 33.5 (May 1986), pp. 533 –541. ISSN: 0098-4094. DOI: 10.1109/TCS.1986.1085953.

[159] **Dávid Tisza**, András Oláh, and János Levendovszky. “Multi-user detection using non-parametric Bayesian estimation by feed forward neural networks”. In: *Telecommunication Systems* (2015), pp. 1–11.

[160] **Dávid Tisza**, András Oláh, and János Levendovszky. “Novel algorithms for quadratic programming by using hypergraph representations”. In: *Wireless personal communications* 77.3 (2014), pp. 2305–2339.

[161] **Dávid Tisza**, Péter Vizi, Janos Levendovszky, and András Oláh. “Multicast Routing in Wireless Sensor Networks with Incomplete Information”. In: *Wireless Conference 2011 - Sustainable Wireless Technologies (European Wireless), 11th European*. 2011, pp. 1–5.

[162] Sergio Verdu. *Multiuser Detection*. Cambridge University Press, 1998. ISBN: 0521593735.

[163] J. Wang. “Discrete Hopfield network combined with estimation of distribution for unconstrained binary quadratic programming problem”. In: *Expert Systems with Applications* 37.8 (2010), pp. 5758–5774.

[164] G. Xia, Z. Tang, Y. Li, and J. Wang. “A binary Hopfield neural network with hysteresis for large crossbar packet-switches”. In: *Neurocomputing* 67 (2005), pp. 417–425.

[165] Minxian Xu, Wenhong Tian, and Rajkumar Buyya. “A Survey on Load Balancing Algorithms for VM Placement in Cloud Computing”. In: *arXiv preprint arXiv:1607.06269* (2016).

[166] Changsheng Yu, Li Yu, Yuan Wu, Yanfei He, and Qun Lu. “Uplink Scheduling and Link Adaptation for Narrowband Internet of Things Systems”. In: *IEEE Access* 5 (2017), pp. 1724–1734.

DOI:10.15774/PPKE.ITK.2019.006

[167] Sun Yunshan, Li Yanqin, Jia Fengmei, Liu Ting, Zhang Liyi, and Zhang Yan. “A novel feed-forward neural network blind multi-user detection algorithm by augmented Lagrange optimization”. In: *Wireless, Mobile and Sensor Networks, 2007. (CCWMSN07). IET Conference on*. 2007, pp. 8–11.