Novel Bit-Loading Algorithms For Spectrum Balancing In Digital Subscriber Lines

by

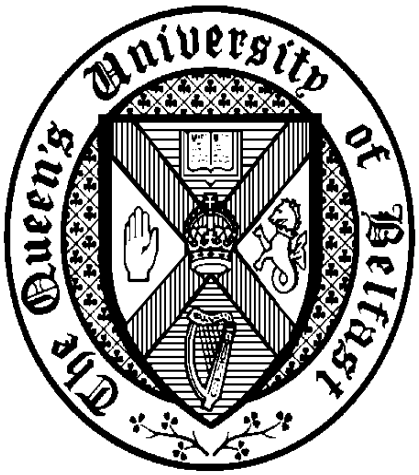
Alastair McKinley

A thesis presented on application for the degree of

Doctor of Philosophy

in the

Faculty of Engineering and Physical Sciences



School of Electronic and Electrical Engineering and

Computer Science

September 2009

Abstract

Digital Subscriber Line (DSL) technologies have been instrumental in delivering mass market broadband access and are currently the dominant technologies utilised in the UK and Europe. In recent years there has been much research focused on improving the performance of Discrete Multi-Tone (DMT) based DSL technologies through more intelligent allocation of power on the DMT subchannels. These techniques are known as Dynamic Spectrum Management (DSM) algorithms. Currently known techniques for DSM generally exhibit either intractable solution times, sub-optimal performance or convergence problems.

There are 3 different levels of DSM operation and this thesis will consider level 2 DSM, in which the spectra are allocated centrally by a Spectrum Management Centre (SMC). There is currently no DSM Level 2 algorithm which can be calculated in a time frame practically useful for realistic DSL bundle sizes (50+ Lines). In this thesis a number of algorithms will be presented which are shown to produce large performance improvements (e.g. 3850 times faster for 7-user ADSL2 Downstream compared to Iterative Spectrum Balancing (ISB)) over existing algorithms in terms of execution speed and also perform very well in terms of achievable data rates (e.g. producing almost equivalent rate regions to ISB for 6-User ADSL2 downstream). The results presented in this thesis show that is it possible (with further enhancements) that Level 2 DSM will be achievable for practical bundle sizes.

The key contributions of this work include a new centralised DSM level 2 algorithm which is shown to achieve performance very close to that of the optimal algorithm (OSB) with equal line weighting. It is shown that this algorithm is

significantly faster than other DSM level 2 algorithms and is in fact tractable for large bundle sizes (tested with up to 48 ADSL lines). Exploitation of computational parallelism within this algorithm is also investigated.

In this thesis it is shown that a greedy level 2 DSM algorithm can achieve near optimal performance for a variety of xDSL networks. It is also shown that it is robust to changing crosstalk models in simulations. Additionally, by noting the behaviour of the new algorithm during the course of its execution, a PSD vector caching technique is developed and shown to improve the speed of the algorithm significantly. A number of new rate search algorithms are derived and investigated, including one which is able to exploit parallel computation. Very large speedups are demonstrated using these new algorithms in comparison to other level 2 DSM algorithms.

Acknowledgements

Completing my PhD studies at Queens has been simultaneously the most challenging and rewarding endeavour I have undertaken. I could not have completed it without the support and help of my family, friends and colleagues.

Firstly, I would like to thank my supervisors Prof. Alan Marshall and Prof. Roger Woods for their outstanding ideas, guidance and good humour.

I would also like to thank all of the members of Prof. Marshall's Advanced Networks Group and some other staff and students for their friendship and advice, with particular mention to, John McGlone, Ben Harrison, Bosheng Zhou, Peter Lee, Shane O'Neill, Simon Cotton and Wenzhe Zhou.

I gratefully acknowledge the financial support of Asidua Ltd.

I would like to reserve very special thanks for my parents, Nuala and Maurice, who have given me tremendous support, particularly when times have been more challenging. I am extremely grateful for all of the opportunities that they have given me.

Lastly, I would like to thank Selina, for her unwavering support throughout my entire PhD.

Notation

Acronyms

ADSL Asynchronous Digital Subscriber Line

ASB Autonomous Spectrum Balancing

AWG American Wire Gauge

AWGN Additive White Gaussian Noise

BBOSB OSB with Branch and Bound

CO Central Office

CP Customer Premises

CP Customer Premises

DMT Discrete Multi-Tone

DSL Digital Subscriber Line

DSM Dynamic Spectrum Management

FDD Frequency Division Duplex

FEXT Far-End Crosstalk

FM Fixed Margin bit-loading

FTTx Fiber to the Home/Cabinet/Neighbourhood

GPGPU General Purpose Computing on Grapics Processing Units

ISB Iterative Spectrum Balancing

ISI Inter Symbol Interference

IWF Iterative Water Filling

LC Levin-Campello

MIPB Multi-User Incremental Power Balancing

NEXT Near-End Crosstalk

OSB Optimal Spectrum Balancing

POTS Plain Old Telephone Service

PSD Power Spectral Density

QAM Quadrature Amplitude Modulation

RA Rate Adaptive bit-loading

RT Remote Terminal

SMC Spectrum Management Centre

SNR Signal-Noise Ratio

SSM Static Spectrum Management

VDSL Very-High Speed Digital Subscriber Line

Symbols

$|H_{FEXT}(f)|^2$ Crosstalk coupling at frequency f

$\bar{p}$ Current average power used on all lines

ϵ Sub-Gradient Step Size

Γ SNR gap

γ_m DMT performance margin

γ_{cg} DMT Coding gain

$\hat{b}$ Maximum number of bits on DMT sub-channel

λ_n Lagrangian dual variable for OSB and ISB

$\sigma_n^2(k)$ Background noise power on line n, tone k

$b(k)$ Bit vector across all lines on tone k

$b_n(k)$ Number of bits on user n, tone k

$h_{jn}(k)$ Channel gain from user j into user n on tone k

N_e Number of nearest neighbours in M-QAM constellation

$p(k)$ Power vector across all lines on tone k

$P(n)$ Current power used on line n

P_e Probability of Error

$p_n(k)$ Transmit power on user n, tone k

P_{budget} DMT Power budget

$Q(x)$ Q Function

R^{tol} Rate tolerance

R_n Data rate on line n

R_n^{target} Rate target on line n

$SNR(k)$ SNR on tone k

w_n Weight of user n

$wp(n)$ Power penalty function on line n

X_{dB} Crosstalk adjustment chosen from Beta model

Contents

List of Figures

List of Tables

List of Algorithms

Chapter 1

Introduction

1.1 DSL Technologies

Since the beginning of the Internet age, new applications have continually demanded more bandwidth.

Recently this trend has been driven by high-definition video streaming and file sharing applications such as bittorrent. Digital Subscriber Lines (DSL) technologies have played an important role in the delivery of higher bandwidth broadband Internet access because of the relatively low cost of deploying DSL networks over existing telephone lines originally used for dial-up access.

Although fibre optic networks are a much more attractive technology in terms of achievable data rates, DSL is still much more prevalent today than fibre due to the significantly lower capital expenditure required. It is predicted that by 2012, only 4% of access connections in Western Europe will be from FTTx, with almost 80% still provided over xDSL[4].

Since their inception, DSL technologies have matured from ADSL (8 Mbits downstream) through to ADSL2/2+ (24 Mbits downstream) and most recently to VDSL2. VDSL2 can deliver up to 100Mbps symmetrically to the customer premises over short loop lengths ($<$ 250m). As the frequency range utilised by the DSL technology is increased (up to 30MHz for VDSL2), the depth of fibre penetration into the access network must be increased to shorten the loop

lengths or the maximum possible speeds will not increase. In this manner, xDSL provides an evolutionary pathway to increased data rates through the progressive deployment of fibre optic cables closer to the customer. It is for these reasons that DSL is likely to be a major provider of broadband Internet access for some time to come.

1.2 Motivation

As DSL technologies utilise the existing copper plant, which was originally designed for transmission in the 0-4kHz band used by the Plain Old Telephone Service (POTS) , they suffer from the limitations of the existing medium. In particular, at the high frequencies used by DSL (up to 30 MHz in VDSL2) there is significant attenuation of the transmitted signal and leakage of signal power into adjacent lines through electromagnetic induction. This leakage of signal power is known as crosstalk.

Crosstalk is a major limiting factor in the achievable data rates of DSL deployments and there is thus far no standard method to mitigate its effects. The modulation scheme used in the most prominent xDSL technologies is Discrete Multi-Tone (DMT). The frequency spectrum is divided up into multiple independent sub-channels which can be individually modulated with a number of bits per symbol based on the Signal-Noise Ratio (SNR). Crosstalk reduces the SNR on the DMT sub-channels in the victim DSL line, reducing the amount of bits per symbol that can be transmitted on that sub-channel at the same error rate.

Over recent years, much research has been undertaken into techniques to reduce the data rate degradations caused by crosstalk. These techniques are known as Dynamic Spectrum Management (DSM). DSM techniques improve the performance in DSL systems by more intelligently allocating signal power on the DMT sub-channels. The purpose of DSM algorithms is to increase the possible data rates in a bundle of DSL lines through mitigation of the effects of crosstalk. DSM techniques may also allow a DSL network to support the same data rates to each user at a lower total power. This is known as “Green DSL” and is currently

an active topic of research in DSM applications [5] [6] [7]. Although it has been demonstrated that DSM techniques can produce large performance gains, many of the currently known algorithms suffer from either intractable solution times or convergence problems.

This thesis concentrates on the design and performance of level 2 DSM algorithms because it is expected that if they can be designed to operate in a tractable time frame, they provide certain unique advantages over level 1 and level 3 techniques. In this case tractable is defined as a time that makes it practically applicable to a real DSL system, on the order of hours or less.

A brief definition of a level 1 DSM algorithm is one which is distributed, that is, DMT sub-channel bit and power assignment is primarily handled by an algorithm within the CP modem with some guidance from a Spectrum Management Centre (SMC). Popular examples of these types of algorithms are Iterative Water-Filling (IWF) [8], Autonomous Spectrum Balancing [9] and Band Preference algorithms [10]. In all of these algorithms, it is required that the modems execute their bit-loading algorithms individually in an iterative fashion. At each iteration, the modem sees the new background noise power generated by the crosstalk from all other modems in the previous iteration. Each modem should approximately converge to the desired performance margin γ_m. This iterative behaviour is illustrated in Figure 1.1.

This distributed, iterative behaviour means that the speed of the DSL network initialisation as a whole is limited by algorithm execution speed on each modem multiplied by the number of iterations required.

In level 3 DSM, crosstalk pre-compensation enables partial or complete cancellation of crosstalk [11]. While this is the ideal case for mitigating crosstalk, it requires extensive deployment of new hardware with significantly increased complexity. It also requires that the signals are co-generated, i.e. on the same hardware. This restricts the market for level 3 DSM solutions to scenarios where it is possible to co-locate modems at the customer premises, for example, at newly constructed shared accomodation. This requirement also makes level 3 DSM difficult and costly to retrofit into existing DSL networks.

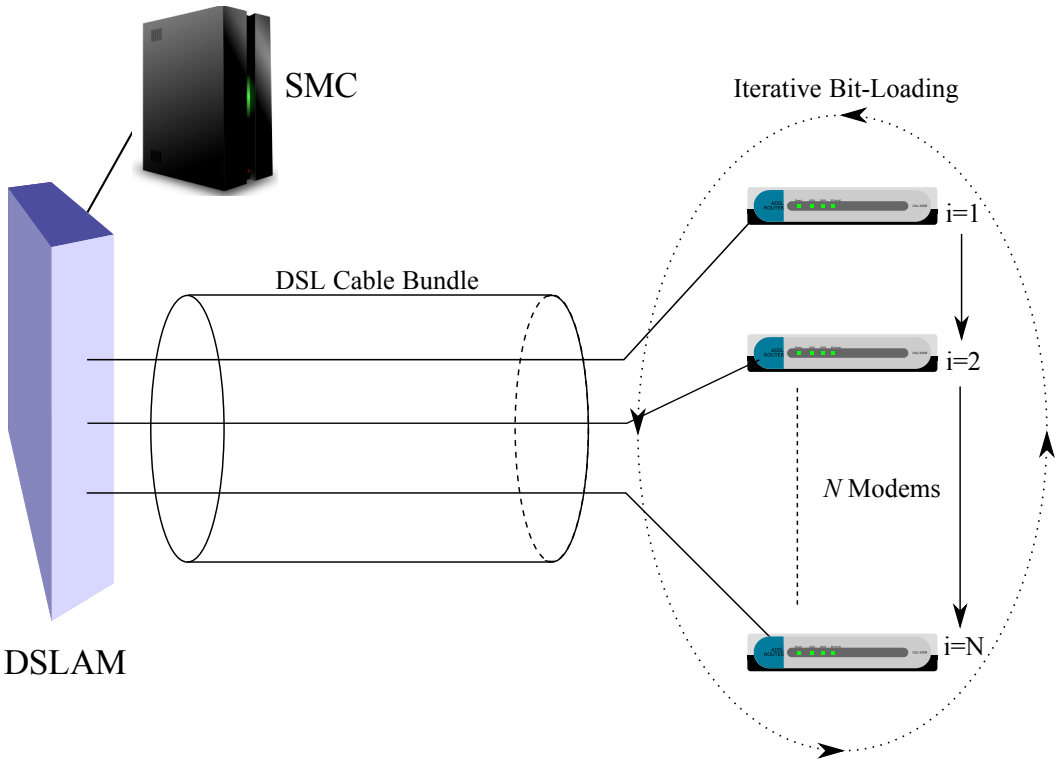
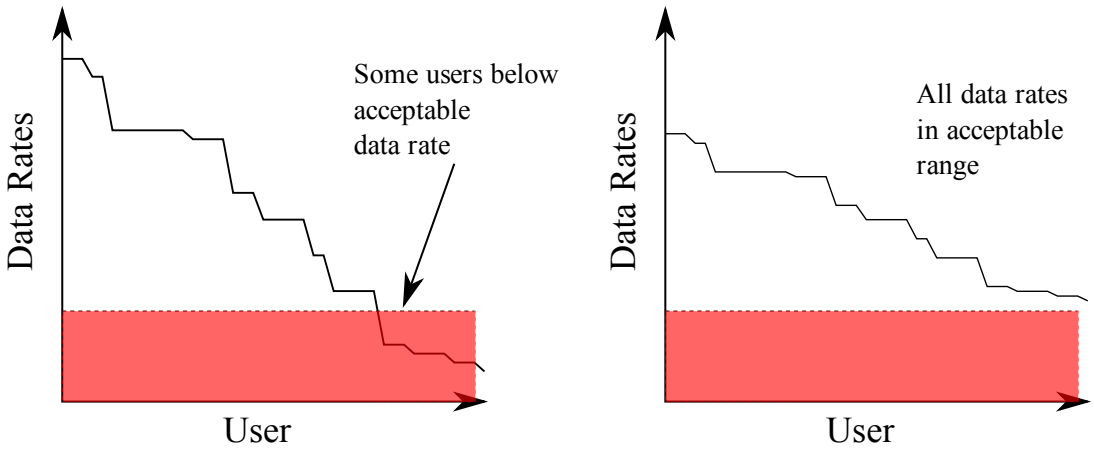


Figure 1.1: Illustration of iterative behaviour in Level 1 DSM operation

In level 2 DSM, all the bit and power allocations are centrally calculated and then distributed to the modems. This approach has several advantages over level 1 and level 3 DSM. Firstly, the performance margins γ_m will be tight to the desired figures when the network is initialised, which is not the case for level 1 DSM. Secondly, there is no settling time for network initialisation as in level 1 DSM, as soon as the bit and power allocations are calculated, the network can begin operation. The limiting factor is now the speed of the level 2 DSM algorithm. Level 2 DSM does not require the addition of new costly and computationally expensive DSP functionality which is required for level 3 DSM.

Figure 1.2 shows an example of the capabilities of a DSM level 2 algorithm in practise. In Figure 1.2(a), the data rates in the bundle are shown in descending order. It may be decided by the network operator that some of these lines are below a minimum level of service, illustrated by the red box. By setting appropriate data rate targets on the lowest data rate lines, a DSM level 2 algorithm can bring these lines up to an acceptable level of service, whilst slightly reducing



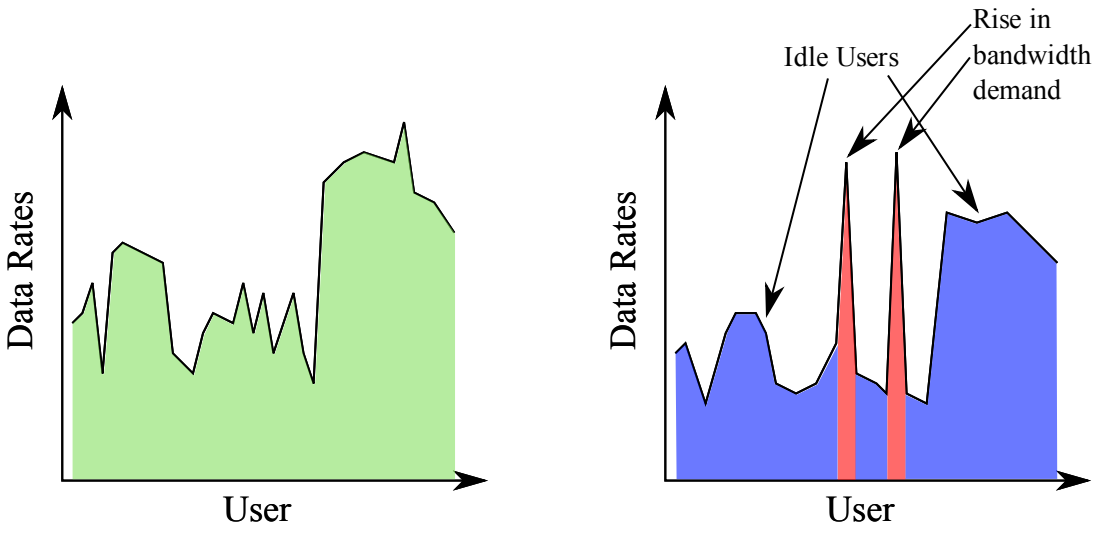
(a) Before DSM

(b) After DSM

Figure 1.2: Data rates ordered from highest to lowest in a DSL bundle before and after DSM is utilised to raise the lowest data rates in the bundle

the data rates of the other lines in the bundle. This is illustrated in Figure 1.2(b). This operation could be completed seamlessly without any service interruption as in level 2 DSM the required spectra are calculated off line in the SMC.

Another possible application of level 2 DSM rate adaption is shown in Figure 1.3. Figure 1.3(a) shows the data rates in the DSL bundle during normal operation. At some point two users in the network are utilising a service (e.g. a large FTP transaction) which could utilise higher bandwidth if it was available. In response to this, the SMC allocates new spectra to all the bundle users which deliver much higher data rates to those users at the expense of the other users' data rates in the bundle. This is shown in Figure 1.3(b). If the other users in the bundle are either idle or have a low bandwidth requirement at that time, there will be no drop in performance perceived by these users. Providing the DSM level 2 algorithm can be executed in a time frame on the order of the typical service time then this rate adaption could be used to further increase the efficiency of the available bandwidth in the entire DSL bundle by constantly adjusting the data



(a) Steady state rates in a DSL network (b) Rate changes after "Real-Time" DSM adaption

Figure 1.3: Data rates before and after a "Real-Time" DSM rate adjustment

rates in the bundle to better match the demand of each user.

The major disadvantage of level 2 DSM is that the channel and crosstalk coupling gains must be known for the algorithm to function. Fortunately, there has been much recent research into crosstalk and channel estimation and it is now possible with good accuracy using slightly modified hardware [12] [13] [14].

At present, known level 2 DSM algorithms are too computationally expensive to be used on realistic DSL bundle sizes (50+ cables) and are merely of theoretical interest. The major theme of this thesis is therefore to find level 2 DSM algorithms which are fast, tractable for large bundle sizes and still provide good data rate performance.

1.3 Thesis Overview and Contributions

Chapter 2 introduces the fundamentals of DMT technology as well as the channel and crosstalk models required for the DSL system model employed in this thesis. The concept of bit-loading in DMT is introduced and the spectrum allocation problem defined. A variety of existing DSM algorithms are presented and their

relative merits discussed.

In Chapter 3, a new DSM level 2 algorithm known as Multi-User Incremental Power Balancing (MIPB) is derived. This is shown through simulation results to achieve performance very close to that of the optimal algorithm when used with "equal line weightings" in a variety of different DSL network configurations. Equal line weightings refers to the values of w_n in the OSB algorithm. When all the values of w_n are set to unity, the algorithm produces a relatively fair allocation of data rates betweens the DSL users. The complexity of the algorithm is investigated and it is found to have a vastly reduced complexity compared to other well known DSM level 2 algorithms. Later in this chapter, the parallelisation of DSM level 2 algorithms including MIPB is investigated which exploit modern multicore hardware to further reduce the execution times of MIPB.

In Chapter 4 of this thesis, it is shown that a greedy algorithm combined with a new rate search algorithm can produce nearly optimal spectrum allocations for a variety of DSL networks. In this case, an optimal solution is defined as a point on the rate region boundary (see Section 2.9.1). A number of different rate search algorithms are derived and simulations are used to examine their complexity. A Power Spectral Density (PSD) vector caching algorithm is proposed and the potential performance improvements are demonstrated through simulation. For a 7 User ADSL2+ network, the best of the new algorithms reaches a solution 3850 times faster on commodity x86 hardware than ISB.

The work presented in chapters 3 and 4 is based on previous work published on greedy algorithms in [15] and [16]. A patent application is currently being pursued for the work in Chapter 4 and will be submitted for publication in the near future.

Finally, in Chapter 5 conclusions are drawn and a section for possible future work is included.

Chapter 2

Background

2.1 Digital Subscriber Lines

Digital Subscriber Lines (DSLs) have become a very common means of providing high speed Internet connectivity to both residential and business customers. In Western Europe there were an estimated 85 million residential broadband connections by the end of 2007 [4]. Approximately 80% of these were provided by DSL.

As DSL is delivered over existing telephone lines, it has been instrumental in enabling the "broadband revolution" as it provides a technology for dramatically increasing bit rates compared to dial-up access, without the need for laying new cables. The original ADSL standard enabled downstream bit-rates of up to 8 Mbits to be obtained on short loops (approx. 1km), which previously only supported 56kbits over a V.92 voice band connection.

As the demand for high bandwidth applications has increased, new DSL technologies have emerged to fulfil the requirements. Upgrading to higher bit-rates is often simply a matter of upgrading the DSL equipment in the Central Office (CO) and Customer Premises (CP), although sometimes loop lengths need to be shortened to achieve significant improvement.

2.2 Transmission Impairments in xDSL

The original telephone loop plant was intended for transmission of voice signals in the 0-4kHz band. DSL systems utilise the spectrum from 25kHz up to 30MHz, depending on the particular flavour of DSL in use. As a result of using the copper medium well outside its original design limits, data transmission in DSL can be problematic.

Signals travelling over the copper telephone lines are generally subject to high levels of attenuation, which increase with increasing frequency. The unshielded copper pairs are also susceptible to noise from various sources including crosstalk from neighbouring copper pairs, radio frequency ingress and impulsive noise. The dominant noise source in DSL is crosstalk from neighbouring lines and is typically at least 10-20dB larger than the background noise [17]. The management of crosstalk in DSL is the main subject of this thesis.

In general, the effects of impulsive noise and RF ingress are not affected by the bit-loading or spectrum balancing techniques investigated in this thesis. Recently, some investigation has been carried out into noise-margin optimisation which may help to mitigate the effects of impulse noise when its characteristics vary with freqeuncy [18]. However, in this thesis, impulse noise and RF ingress are considered to be part of the background noise and only the nature and effects of crosstalk noise are considered.

2.2.1 Crosstalk

Crosstalk noise is caused by leakage of signal power from one DSL line into another. The coupling between neighbouring DSL lines is caused by electromagnetic induction and occurs where DSL cables overlap within a bundle of DSL cables. The signals in a particular line create an electromagnetic field which in turn induces a current in neighbouring lines.

There are two major types of crosstalk known as Near-End Crosstalk (NEXT) and Far-End Crosstalk (FEXT), which are illustrated in Figure 2.2.1. NEXT is caused by leakage of a user's signal into a receiver at the same end as the

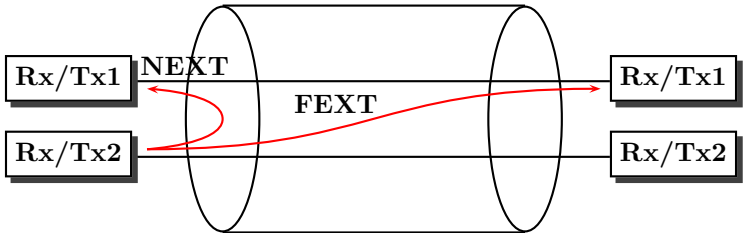


Figure 2.1: NEXT and FEXT in DSL

transmitter. Conversely, FEXT is caused by leakage of of signal power from a transmitter at one end of a DSL into a receiver on the other side.

In the majority of DSL systems, NEXT can be eliminated by employing Frequency-Division-Duplexing (FDD). In FDD, the frequency bands used for downstream and upstream transmission are separated.

The magnitude of crosstalk coupling varies greatly with frequency but is typically stationary over time. It is possible that the coupling functions may change in areas with large changes in humidity or temperature but these effects are usually negligible.

2.3 DSL Channel Models

Throughout this thesis, models for both the direct channel and crosstalk gains will be utilised in order to simulate a variety of DSL network configurations. Models are utilised rather than real data because measurement of these values is at present a time consuming process, although there is a large effort currently being undertaken to standardise the measurement process for the purposes of DSM.

The direct gain channel models used in this thesis are calculated by the long line approximation for the channel shown in [19]. The approximation is that for a long DSL line the input line impedance V_1 is equal to the characteristic impedance V_0. Given this assumption, for a line of length d with characteristic impedance Z_0, load impedance Z_l, source impedance Z_s and propagation constant γ, the voltage transfer function is given by

$$H = \frac{Z_0 \cdot sech(\gamma d)}{Z_s \cdot [\frac{Z_0}{Z_l} + tanh(\gamma d)] + Z_0 \cdot [1 + \frac{Z_0}{Z_l} \cdot tanh(\gamma d)]} \tag{2.1}$$

The characteristic impedance Z_0 of the DSL line is given by the relationship

$$Z_0 = \sqrt{\frac{Z}{Y}} \tag{2.2}$$

Where Z and Y are the impedance and admittance per unit length respectively.

The value of the propagation constant γ is given by

$$\gamma = \sqrt{Z \cdot Y} \tag{2.3}$$

2.3.1 RLCG Parameter Curve Fitting

In order to calculate the frequency dependent values of R,C,L and G for a particular length of DSL line, parameterised models are used to ensure the models follow smooth curves with frequency [19]. These models are particularly useful as practical measurements suffer from relatively large margins of error.

The parameterised model for $R(f)$ is expressed as

$$R(f) = \frac{1}{\frac{1}{\sqrt[4]{r_{0c}^4 + a_c \cdot f^2}} + \frac{1}{\sqrt[4]{r_{0s}^4 + a_s \cdot f^2}}} \tag{2.4}$$

where r_{0c} is the copper DC resistance and r_{0s} is the steel resistance. The values a_c and a_s are constants which characterise the increase in resistance with frequency due to the skin effect.

The smoothed model for $L(f)$ is as follows

$$L(f) = \frac{l_0 + l_\infty (\frac{f}{f_m})^b}{1 + (\frac{f}{f_m})^b} \tag{2.5}$$

where l_0 and l_∞ are the low and high frequency inductance respectively and the values b and f_m are chosen to characterise the transition from low to high frequencies.

For the smoothed capacitance $C(f)$, the model is

$$C(f) = c_\infty + c_0 \cdot f^{-c_e} \tag{2.6}$$

Finally, the smoothed conductance $G(f)$ is given by

$$G(f) = g_0 \cdot f^{g_e} \tag{2.7}$$

Throughout this thesis, the majority of results will be calculated using models for AWG 24 wires. Table 2.1 shows the parameters used to model the RCLG parameters for AWG 24. Parameter values for various cable types are available in [20]. For the purposes of analsying DSM algorithms, the specific cable model used is not particularly important, as long as the same model is used for all comparisons.

Figure 2.2 shows the gain over the ADSL frequency range for different lengths of AWG 24 wire given by the smoothed RCLG models, when these values are used to calculate the transfer functions given by equation 2.1.

2.4 Crosstalk Models

The basis for the crosstalk models used in this thesis are the 1% worst case crosstalk models from [19]. The 1% worst case models produce crosstalk coupling functions that follow smooth curves with frequency. Although this is not generally

r_{0c}	r_{0s}	a_c	a_s
174.55888 ohms/km	∞ ohms/km	0.0530734	0.0
l_0	l_∞	b	f_m
617.295 μH/km	478.97 μH/km	1.1529	553.76 kHz
c_∞	c_0	c_e	
50 nF/km	0.0 nF/km	0.0	
g_0	g_e		
234.874 fS/km	1.38		

Table 2.1: Characteristics of AWG 24 UTP [1]

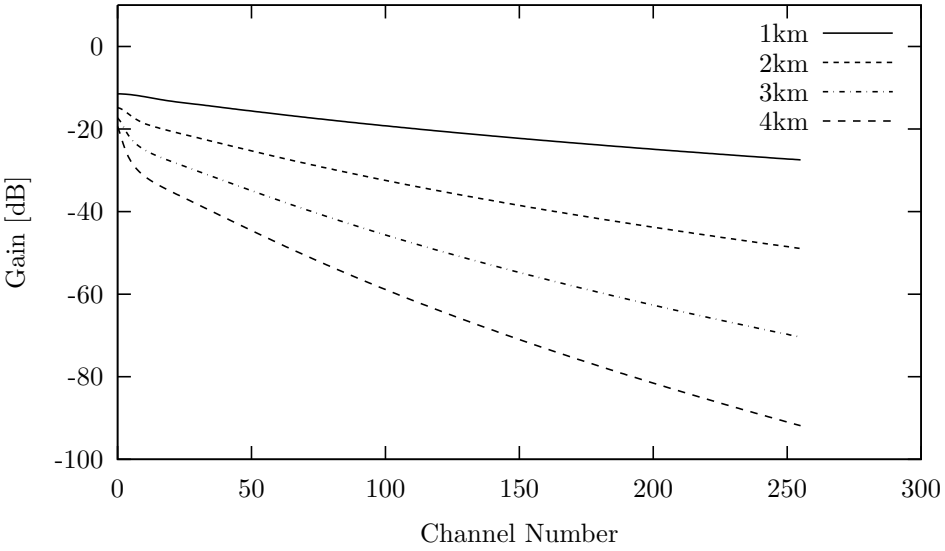


Figure 2.2: Gain vs DSL channel number for varying lengths of AWG 24 UTP

the case in practise, these models ensure that the calculated crosstalk coupling will be worse than in 99% of real life twisted pair scenarios.

2.4.1 FEXT Modelling

The simplified 1% worst case ETSI FEXT model [21] is given by:

$$|H_{FEXT}(f)|^2 = N^{0.6} K_{FEXT} f^2 |H_{channel}(f, L)|^2 \tag{2.8}$$

Where f is the frequency and L is the length of overlapping cable in the bundle. In a practical DSL system, the number of cables that overlap and their overlap distance vary along the length of the cable between the Central Office (CO) end and the Customer Premises (CP). To model this, equation 2.8 is used in conjunction with the rules in [22] to calculate the crosstalk coupling functions in a more complex scenario such as a mixed CO and Remote Terminal (RT) deployment.

2.4.2 Statistical Crosstalk Model

The 1% worst case model tends to over-estimate the crosstalk coupling functions between DSL lines [18]. Additionally it assumes that the crosstalk coupling functions are independent of the relative positions of the cables in the bundle.

In order the improve the realism of the 1% worst case model, it was recently extended with a statistical approach [23]. This model was developed by the DSL industry through the analysis of raw FEXT coupling data measured in a 100x100 DSL binder [24] consisting of four sub-binders. From the model deduced from the coupling data, a adjustment is applied to the 1% worst case model to produce more realistic FEXT coupling values. The modified crosstalk coupling function between any two lines j and k is now given by:

$$|H_{FEXT}^{j,k}(f)|^2 = |H_{FEXT}^{j,k}(f)|^2 \times 10^{\frac{X_{dB}}{10}} \tag{2.9}$$

Where the value X_{dB} is a value picked from a Beta distribution. A random draw from this distribution for a 100 pair DSL bundle is made available in [25].

2.5 Discrete Multi-Tone Modulation

The modulation scheme that has been standardised for DSL is Discrete Multi-Tone (DMT). It was first described in [26] and [27]. DMT is designed to be able to operate over channels with significantly different channel and noise characteristics over the transmission bandwidth, such as those encountered in DSL systems.

In DMT, the channel bandwidth is divided up into a set of independent sub-channels also known as tones. The sub-channel bandwidth is chosen to be sufficiently narrow so that the transfer function is approximately flat across its bandwidth. This means that each sub-channel is free from ISI and thus complex time domain equalisation is not required. Figure 2.3 illustrates the division of channels in a FDD DMT system.

Each DMT sub-channel can be viewed as an independently M-QAM modu-

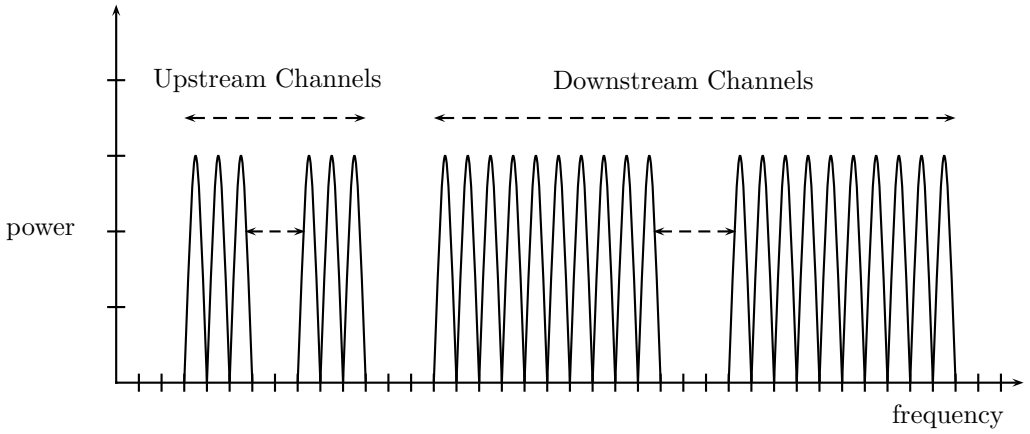


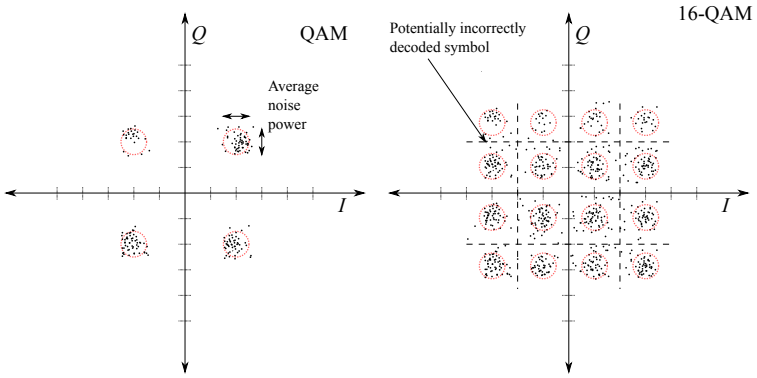
Figure 2.3: Illustration of a Frequency Division Duplex DMT system

lated channel for data transmission. The number of bits loaded on each DMT sub-channel is determined by the constellation size. The symbol error probability P_e is dependent on the constellation size and the power allocated to that sub channel. This is roughly illustrated in Figure 2.4. Figure 2.4(a) shows a square QAM constellation, which encodes 2 bits per symbol. The red circles approximately represent the average noise power and the dots represent detection of received symbols which all fall within the correct detection windows, resulting in error free transmission. In Figure 2.4(b), the result of increasing the constellation size to 16-QAM is illustrated. The transmit signal power and noise power remain unchanged but because of the smaller detection windows, some of these symbols may be decoded incorrectly, resulting in symbol errors. This is remedied in Figure 2.4(c) which illustrates the result of increasing the transmit signal power on this tone. The signal-noise ratio is increased and all of the decoded symbols fall within the correct detection windows resulting in error free transmission.

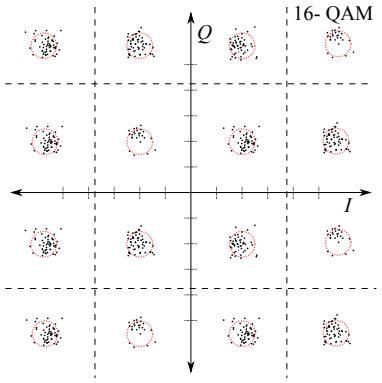
The process of assigning the number of bits per symbol (constellation size) and power to each DMT sub-channel is known as "bit-loading".

2.5.1 Bit-Loading

In DMT systems, the process of assigning bits to individual sub-channels is known as "bit-loading". When considering a single DMT user, the maximum capacity



(a) 4-QAM (b) 16-QAM with low SNR



(c) 16-QAM with high SNR

Figure 2.4: Illustration of the relationship between constellation size, noise power and symbol error probability for M-QAM

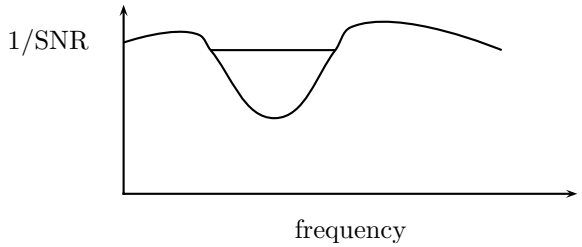


Figure 2.5: Illustration of the Water-Filling concept. The depth of the water level indicates the amount of energy allocated at that frequency

of a system with a fixed power budget is well known. It is given by the so called "Water-Filling" solution first introduced by Claude Shannon in 1948 [28]. The water filling solution is so named because the mathematical solution resembles the concept of "pouring" energy in the channels with the best Signal-Noise-Ratio (SNR). This is illustrated in Figure 2.5 which depicts the inverse signal to noise ratio graphed against frequency.

In an ideal system, the waterfilling solution gives the maximum capacity of the system. A practical communications system must implement a data rate lower than capacity to be reliable. The so called "SNR-gap" or "Shannon-gap" is a measure of the gap from capacity for a communications system. The derivation for a DSL system based on M-QAM is shown in the following Section [18].

2.5.2 The SNR-Gap Approximation

The Shannon capacity of an Additive White Gaussian [1] Noise (AWGN) channel is given by the following formula:

$$C(k) = \log_2(1 + SNR(k)) \tag{2.10}$$

This is the maximum rate at which error free transmission is possible, given unlimited coding complexity and decoding delay. Assuming that a DSL system utilises an uncoded square M-QAM constellation, the probability of a symbol

[1] Crosstalk approaches gaussian behaviour as the number of crosstalkers increases, the central limit theorem loosely applies [29]

error on a particular sub-channel is given by

$$P_e \approx N_e Q\left(\sqrt{\frac{3}{M-1}SNR}\right) \tag{2.11}$$

Where the $Q(x)$ function gives the probability that a Gaussian variable with a unit mean and variance exceeds the value x and N_e is the number of nearest neighbours in the M-QAM constellation.

$$Q(x) = \int_x^\infty \frac{1}{\sqrt{2\pi}} e^{-u^2/2} du \tag{2.12}$$

The $Q(x)$ function can also be written in terms of the the standard error function $erf(x)$.

$$Q(x) = \frac{1}{2}\left(1 - \text{erf}\left(\frac{x}{\sqrt{2}}\right)\right) \tag{2.13}$$

$$Q^{-1}(x) = \sqrt{2}\,\text{erf}^{-1}(1-2x) \tag{2.14}$$

The number of bits that can encoded by an M-QAM constellation is given by

$$b = \log_2(M) \tag{2.15}$$

By rearranging equation 2.11 for M, we obtain the following expression for the number of bits encoded on a sub-channel

$$b = \log_2\left(1 + \frac{SNR}{\frac{1}{3}\left(Q^{-1}(\frac{P_e}{N_e})\right)^2}\right) \tag{2.16}$$

By comparing this expression with equation 2.10, we can define the uncoded channel gap $\Gamma_{uncoded}$ as

$$\Gamma_{uncoded} = \frac{1}{3}\left(Q^{-1}(\frac{P_e}{N_e})\right)^2 \tag{2.17}$$

Thus, the equation for the number of bits that can be loaded on a sub-channel is given by

$$b = \log_2\left(1 + \frac{SNR}{\Gamma_{uncoded}}\right) \tag{2.18}$$

The SNR-gap Γ represents the distance from the theoretical maximum capacity at a particular symbol error probability. In a practical system, a performance

margin is employed to ensure the system always performs at least as well as its designed symbol error probability under unforeseen noise conditions.

The SNR-gap can also be reduced by employing forward error correction. The reduction in gap is called the coding gain γ_{cg}. The resulting complete formula for Γ is shown below.

$$\Gamma = \Gamma_{uncoded} + \gamma_m - \gamma_{cg} \tag{2.19}$$

In a typical DSL system used for data transmission, the target symbol error probability is set at $1e^{-7}$ which corresponds to an uncoded gap of $9.8dB$ as calculated by equation 2.17. Typical values of γ_{cg} and γ_m are $3dB$ and $6dB$ respectively which results in a value of $\Gamma = 12.8dB$ at $P_e = 1e^{-7}$.

2.6 DSL System Model

In this section, the system of equations which describe a multi-user DSL system will be derived. In essence, this system generates the vector of Power Spectral Densities (PSDs) on each line which are required to support a particular vector of bits on that tone, with a given performance margin.

It is assumed in this thesis that crosstalk coupling happens on a per tone basis, which is true when DMT blocks are received synchronously or when "Zipper" [30] DMT is used. Zipper DMT adds a cyclic suffix and pulse shaping to the transmitted DMT frames along with windowing of the received signal to minimise the sidelobes of the received signal, reducing out of band leakage to a negligible level.

In a multi-user DSL system with N users and K tones with FEXT coupling, the achievable bit-loading on line n tone k is as follows [31]:

$$b_n(k) = \log_2 \left(1 + \frac{p_n(k)|h_{nn}(k)|^2}{\Gamma \left(\sigma_n^2(k) + \sum_{j \neq n}^{N} p_j(k)|h_{jn}(k)|^2 \right)} \right) \tag{2.20}$$

Re-arranging the formula (2.20) letting $f(b_n(k)) = \Gamma(2^{b_n(k)} - 1)$ we obtain:

$$p_n(k) - f(b_n(k)) \sum_{j \neq n}^{N} p_j(k) \frac{|h_{jn}(k)|^2}{|h_{nn}(k)|^2} = f(b_n(k)) \frac{\sigma_n^2(k)}{|h_{nn}(k)|^2} \tag{2.21}$$

For a particular tone k, equation (2.21) is an N-dimensional linear system of equations, where N is the number of lines. This can be written in matrix form:

$$A(k)P(k) = B(k) \tag{2.22}$$

where

$$A(k)_{ij} = \begin{cases} 1, & \text{for } i = j \\ \frac{-f(b_i(k))|h_{ji}|^2}{|h_{ii}|^2}, & \text{for } i \neq j \end{cases} \tag{2.23}$$

$$P(k) = [p_1(k) \dots p_i(k) \dots p_N(k)]^T \tag{2.24}$$

$$B(k) = \left[\frac{f(b_1(k))\sigma_1^2}{|h_{11}|^2} \dots \frac{f(b_i(k))\sigma_i^2}{|h_{ii}|^2} \dots \frac{f(b_N(k))\sigma_N^2}{|h_{NN}|^2} \right]^T \tag{2.25}$$

This can be solved for $P(k)$ via direct inversion or by LU/QR decomposition. $P(k)$ is a vector which contains the power value required on each line to support a vector of bits $b(k)$ on tone k. (Note $B(k)$ is a function of $b(k)$).

2.7 Discrete Waterfilling

The waterfilling algorithm provides the optimal solution to maximise the capacity of the channel for a single user with a continuously variable bit distribution. In a real communications system, each channel must transmit an integer number of bits. When the waterfilling analogy is adapted for a real DMT system, it is known as discrete waterfilling.

There are two variations of the discrete waterfilling problem, rate-adaptive (RA) and fixed-margin (FM). In RA mode, the objective is to maximise the bit-rate for a given power budget. Formally, the single-user RA problem is stated as follows

$$max \;\; R = \sum_{k=1}^{K} b(k) \tag{2.26}$$

$$s.t \;\; \sum_{k=1}^{K} p(k) \;\; \leq P_{budget} \tag{2.27}$$

Where k is the tone index and K is the number of tones.

In FM mode, the objective is to minimise the energy required to transmit at a particular bit rate. The single-user FM problem is stated as follows

$$min \quad \sum_{k=1}^{N} p(k) \tag{2.28}$$

$$s.t \quad \sum_{k=1}^{N} b(k) \;\; \geq B_{target} \tag{2.29}$$

The most widely used discrete waterfilling algorithm is known as the Levin-Campello (LC) Algorithm [32].

The solution to the single-user RA and FM modes is surprisingly simple. In RA mode, the LC algorithm adds a bit to that tone which requires the least energy to do so until the output is equal to the power budget. In FM mode, the algorithm adds bits the lowest energy tone until the target data rate is met.

The RA LC algorithm is shown is Algorithm 1. $|H(k)|$ is the gain of channel k and $\sigma(k)$ is the background noise power on channel k. The FM version of the LC algorithm is shown in Algorithm 2.

Algorithm 1 Levin-Campello Algorithm in Rate Adaptive Mode

repeat
  $\text{argmin}_k \; \frac{\Gamma(2^{b(k)}-1)\sigma(k)}{|H(k)|} - \frac{\Gamma(2^{b(k)+1}-1)\sigma(k)}{|H(k)|}$
  Increment $b(k)$
until $\sum p(k) == P_{budget}$

Algorithm 2 Levin-Campello Algorithm in Fixed Margin Mode

repeat

argmin$_k$ $\frac{\Gamma(2^{b(k)}-1)\sigma(k)}{|H(k)|} - \frac{\Gamma(2^{b(k)+1}-1)\sigma(k)}{|H(k)|}$

Increment $b(k)$

until $\sum b(k) == B_{target}$

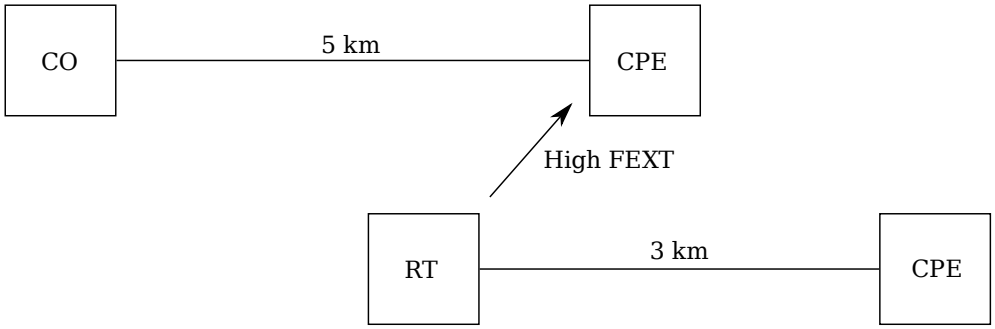


Figure 2.6: A two user DSL network which exhibits the near-far effect in the downstream direction

2.8 Spectrum Management

Although the discrete waterfilling algorithm provides the optimal solution for a single user DSL system, it performs very sub-optimally in a multi-user scenario. In a real DSL system a number of users (typically 50-100 in a bundle) will generate crosstalk into one another. If each modem executes a LC algorithm independently, the overall performance of the whole bundle will be sub-optimal. The level of sub-optimality depends on the network configuration. A particularly bad case is shown in Figure 2.6.

In the downstream direction, the modem in the RT causes a large amount of crosstalk into the CO line. As the CO line is long, it already has a limited achievable data rate due to attenuation. The large amount of crosstalk from the CO line causes a significant reduction in its performance.

To improve this situation, techniques known as Spectrum Management can be employed. The early forms of Spectrum Management are now known as Static Spectrum Management (SSM) methods. They are quite simplistic and involve

setting limits on the transmit PSD in certain bands. SSM techniques help improve performance somewhat but are now considered obsolete in favour of more complex methods known as Dynamic Spectrum Management (DSM)[33] which is now a very active research topic [8] [17] [34] [9] [10] [35] [2].

2.9 Dynamic Spectrum Management

A DSL system employing Dynamic Spectrum Management (DSM) will utilise a Spectrum Management Centre (SMC) which monitors line conditions. The level of monitoring required depends on the type of DSM in use. The SMC will attempt to optimise the performance of the DSL system based on the conditions in the bundle. It may adjust parameters such as margin, power budget, power-spectral-density (PSD) masks and target data rates, depending on the DSM level employed by the SMC.

In essence, a DSM technique will attempt to solve the Spectrum Balancing problem and achieve a result as close to the optimum as possible. The Spectrum Balancing problem for a 2-user situation can be stated as shown in equation 2.30.

$$\max R_2 \quad \text{s. t.} \quad R_1 \geq R_1^{target} \tag{2.30}$$

More generally, the spectrum management problem is a maximisation of a weighted sum of the data rates in a bundle, where the weights are chosen to select the data rates of each user. The full expression for the weighted sum is shown in equation 2.31.

$$\max \sum_{n=1}^{N} w_n \sum_{k=1}^{K} \log_2 \left(1 + \frac{p_n(k)|h_{nn}(k)|^2}{\Gamma \left(\sigma_n^2(k) + \sum_{m \neq n}^{N} p_m(k)|h_{jm}(k)|^2 \right)} \right) \tag{2.31}$$

$$\text{s. t.} \sum_{k=1}^{K} p_n(k) \leq P_n \forall n \tag{2.32}$$

DSM techniques are divided into four categories, which are illustrated in Table 2.2. Each new level increases the achievable performance gains over the previous level, at the cost of increased co-ordination between DSL lines.

Level 0 DSM means that no DSM is employed and each modem will operate completely autonomously, which is the way traditional DSL systems operate. In level 1 DSM, an SMC will adjust DSL lines parameters individually, in order to improve performance and reduce crosstalk. This usually means adjusting the parameters of the discrete water-filling Algorithm [32] such as power budget. Iterative waterfilling [8] and Autonomous Spectrum Balancing (ASB) [9] are examples of level 1 DSM techniques.

In level 2 DSM, the spectra of managed DSL lines are jointly optimised and allocated by the SMC. This requires knowledge of the crosstalk coupling gains between the individual lines in the bundle. There are many examples of level 2 DSM algorithms developed in literature. The optimal performance is given by Optimal Spectrum Balancing (OSB) [17].

Iterative Spectrum Balancing [34] is based on the same framework as OSB with modifications to significantly reduce it's complexity. SCALE [35] is a spectrum balancing algorithm based on a successive convex relaxation of the original spectrum balancing problem. It achieves very close to the optimal result with a significantly lower complexity than OSB. It is not entirely clear at this stage precisely how it compares to OSB and ISB, as their implementation and complexity analysis are non-trivial and to date there has been no detailed analysis of it's complexity published.

Level 3 DSM requires signal level co-ordination of DSL modems. This is known as "vectored" [11] transmission. With vectored transmission, crosstalk can be, in principle, effectively cancelled. This results in the highest possible performance of the DSL bundle. Level 3 DSM is less flexible than level 1 and level 2 DSM as it requires significantly modified hardware with increased cost, due to computationally complex nature of crosstalk pre-compensation. It also necessitates physical co-location of transmitting modems which is particularly problematic in the upstream direction.

Table 2.2: Table of Dynamic Spectrum Management Levels

DSM Level	Description
0	No DSM
1	Some control of individual lines bit loading parameters
2	Joint spectral optimisation of multiple lines
3	Signal level co-ordination between multiple lines

2.9.1 Rate Regions

A rate region illustrates the region of possible rate vectors for a particular DSL network and spectrum balancing algorithm. In general, the larger the rate region that can be traced by a spectrum balancing algorithm, the better its performance.

Figure 2.7 shows a rate region for a two user DSL network with two potential rate vectors marked. The solid and dashed lines represent two different spectrum balancing algorithms. From this it can be deduced that the algorithm which traces the solid line performs better in terms of possible data rates than the algorithm which traces the dashed line.

Rate regions like those shown in Figure 2.7 can only be visualised in two dimensions. With more than two lines, it becomes very difficult or perhaps intractable to trace the entire rate region. When there are more than two lines, the usual method of analysis is to fix the rates of all other lines in the bundle and draw the rate region for the two remaining lines.

2.9.2 Iterative Water-Filling

IWF was one of the early developments in DSM and showed that large performance gains were possible over traditional SSM methods. IWF has many advantageous features as a DSM algorithm such as low-complexity, de-centralised control and relative ease of implementation. IWF utilises a traditional discrete water-filling algorithm on each modem such as the Levin-Campello Algorithm [32] with some parameters of the algorithm controlled by the SMC.

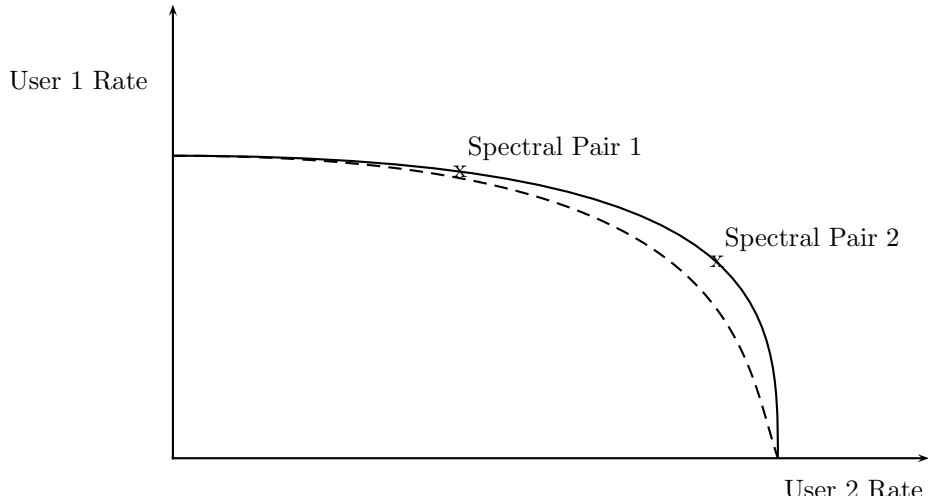


Figure 2.7: Two User Rate Region with two different spectral pairs shown

The basic idea of the algorithm is to iteratively adjust the power constraints of each modem so that they use just enough power to obtain their target data rate. To a certain extent, this helps to protect modems from the so called "near-far" effect which is illustrated in Figure 2.6.

In the downstream direction, the modem in the RT causes a large amount of crosstalk into the CO line. As the CO line is long, it already has a limited achievable data rate due to attenuation. The large amount of crosstalk from the CO line causes a significant reduction in its performance. This is a classic scenario where any form of DSM will produce a large increase in performance in comparison to SSM. Even though IWF provides a performance increase here, it is actually highly suboptimal in this scenario as other DSM techniques can outperform IWF greatly. The algorithm for IWF is shown in Algorithm 3.

Another problem with IWF is that it has been shown that the performance margins γ_m on each tone will not necessarily converge. In Section 3.4.3 two scenarios are shown in which margin convergence does not occur.

2.9.3 Optimal Spectrum Balancing

The optimal solution to the spectrum balancing problem was found by Cendrillon et al. in 2004 [17] and is known as the Optimal Spectrum Balancing (OSB)

Algorithm 3 Iterative Water-Filling

```
repeat
    for n = 1...N do
        Execute LC algorithm with power budget P_n on line n
        if R_n > R_n^target then
            P_n = P_n − δ
        else
            P_n = P_n + δ
        end if
    end for
until convergence
```

algorithm. Their solution is based on a Lagrangian dual decomposition of the spectrum balancing problem given in equation 2.31.

The innovation is to form the Lagrangian dual problem of the optimisation in equation 2.31.

$$L = \sum_{n=1}^{N} w_n \sum_{k=1}^{K} \log_2 \left(1 + \frac{p_n(k)|h_{nn}(k)|^2}{\Gamma \left(\sigma_n^2(k) + \sum_{m \neq n}^{N} p_m(k)|h_{jm}(k)|^2 \right)} \right) - \lambda_n \left(\sum_{k=1}^{K} p_n(k) \right) \tag{2.33}$$

The Lagrangian multipliers are chosen so that the power constraints on each line are tight. It was shown by Cendrillon et al. in [17] that the unconstrained maximisation of the Lagrangian is approximately equivalent to the constrained maximisation of the original problem. This was generalised by Yu and Lui in [36] who also showed that this approximation is exact for multicarrier systems with a large number of channels.

By examining equation 2.33 it can be seen that the Lagrangian can be decoupled on each tone to form the per-tone Lagrangian shown in equation 2.34. Thus by optimising the per tone Lagrangian and searching the λ space the original spectrum balancing problem can be solved. The lambda values can be found by bisection or more efficiently via sub-gradient or ellipsoid methods.

Users	Execution Time
2	95.14 Seconds
3	27.93 Minutes
4	7.97 Hours
5	5.95 Days
6	110 Days
7	4.85 Years (Projected)
8	82.6 Years (Projected)

Table 2.3: OSB execution times for Downstream ADSL as listed in [2]

$$L(k) = \sum_{n=1}^{N} w_n b_n(k) - \sum_{n=1}^{N} \lambda_n p_n(k) \tag{2.34}$$

Unfortunately, the per-tone Lagrangian is not convex, so in order to find the optimum solution, an exhaustive search over all possible bit combinations is required. The number of possible bit combinations on each tone is given by $(b_{max} + 1)^N$. This means that the per-tone optimisation and hence the OSB algorithm itself is exponential in the number of users N.

This is the most serious disadvantage of OSB, as it is intractable for more than 4-5 users as illustrated in Table 2.3. The OSB algorithm utilising a sub-gradient λ update method is shown in Algorithm 4.

Algorithm 4 Optimal Spectrum Balancing

repeat
  repeat
    for $k = 1 \dots K$ **do**
      $\text{argmax}_{b(k)} L(k) = \sum_{n=1}^{N} w_n b_n(k) - \sum_{n=1}^{N} \lambda_n p_n(k)$
      (Solve by N-d exhaustive search)
    end for
    $\lambda_n = \lambda_n + \epsilon(\sum_{k=1}^{K} p_n(k) - P_n)$
  until convergence
  $w_n = w_n + \epsilon(\sum_{k=1}^{K} b_n(k) - R_n^{target})$
until convergence

2.9.4 Iterative Spectrum Balancing

As the complexity of OSB makes the algorithm intractable for even a moderate number of users, Iterative Spectrum Balancing (ISB) was developed by Cendrillon and Moonen [34] in an attempt to address this problem. It was later more thoroughly investigated by Yu and Lui in [36]. ISB is based around the same Lagrangian decomposition as OSB, the difference being the method of maximising the per tone Lagrangian. Instead of an exhaustive search over all bit-loading combinations, the Lagrangian is optimised on each line in turn. In OSB, the optimisation is a "grid search" whereas in ISB the optimisation is an iterative "line search". This iterative line search often leads to the global optimum, but it not guaranteed to do so [34]. In practise the results are similar to OSB, with a slight performance reduction. ISB achieves a significant complexity reduction compared to OSB. In terms of the number of users N, OSB is $O(2^N)$ where ISB is $O(N^2)$. This means that ISB can optimise spectra for entire bundles of 50-100 lines. The ISB algorithm is shown in Algorithm 5.

Algorithm 5 Iterative Spectrum Balancing

repeat
  for $k = 1 \dots K$ **do**
    repeat
      for $n = 1 \dots N$ **do**
        Fix $b_m(k) \forall m \neq n$
        $\text{argmax}_{b(k)} L(k) = \sum_{n=1}^{N} w_n b_n(k) - \sum_{n=1}^{N} \lambda_n p_n(k)$
        (Solve by 1-d exhaustive search)
      end for
      $\lambda_n = \lambda_n + \epsilon(\sum_{k=1}^{K} p_n(k) - P_n)$
    until convergence
  end for
  $w_n = w_n + \epsilon(\sum_{k=1}^{K} b_n(k) - R_n^{target})$
until convergence

2.9.5 Optimal Spectrum Balancing With Branch and Bound

OSB with Branch and Bound [2] (BBOSB) is another spectrum balancing algorithm based on the Lagrangian dual decomposition. The key difference between OSB and BBOSB is the method of solving the per tone Lagrangian in equation

2.34, which is solved more efficiently with a branch and bound procedure rather than an exhaustive search.

Branch and bound is a generic technique for solving optimisation problems which consists of a branch operation and a bounding operation. The branching operation splits the original solution set into smaller sub regions. The bounding operation then finds the upper and lower bound of each sub regions. Any region which has an upper bound lower than the current maximum lower bound can be safely discarded from the search. This can have a significant effect on the complexity of the algorithm. In [2] the authors note that the BBOSB algorithm calculates the optimum spectra for an eight line scenario in one four thousandth of the execution time of OSB.

Although BBOSB is much more efficient than OSB, it is still exponential in the number of users N. However, where OSB is only tractable for 4-5 lines, BBOSB is tractable up to approximately 10 lines. This fact makes the BBOSB algorithm very useful for testing the performance of other spectrum balancing algorithms for moderate size scenarios. The branch and bound procedure is shown in Algorithm 6.

Algorithm 6 Branch and Bound Procedure for OSB

repeat
  repeat
    for $k = 1 \dots K$ **do**
      Start with initial region which contains all solutions
      repeat
        Split each region in half in each dimension into 2^N sub regions
        Calculate Upper and Lower Bound of each sub region
        Discard regions with an upper bound lower than max_low
      until Region converges to one single point
    end for
    $\lambda_n = \lambda_n + \epsilon(\sum_{k=1}^{K} p_n(k) - P_n)$
  until convergence
  $w_n = w_n + \epsilon(\sum_{k=1}^{K} b_n(k) - R_n^{target})$
until convergence

2.9.5.1 Efficient Lambda Update Methods

One of the problems with Lagrangian dual based spectrum balancing methods is to select the appropriate update method for λ. The most reliable method is bisection of each individual λ_n in turn, but this is unacceptably slow to converge.

The sub-gradient method is the most straightforward multi-user λ_n update method but includes the problem of selecting an appropriate step size. Choosing a step size that is too low will result in a very slow convergence time. Choosing a step size that is too large may result in oscillation of the solution about the power constraints resulting in a failure to converge.

The authors in [37] presented a λ_n update method utilises a gradient search with a dynamic step size. Briefly stated, the step size is increased as long as the Euclidean distance of the current solution from the system power constraints decreases. When it is discovered that the Euclidean distance is increasing, a new search is initiated started from the closest point that has currently been reached. This algorithm is illustrated in Algorithm 7. This algorithm is utilised in many of the results obtained in this thesis, however it is noted that there are often problems with this approach resulting from a tendency to get stuck in local minima in the power plane.

Algorithm 7 Efficient Lagrangian Update Method

while $|P(n) - P_{budget}| < p_{tolerance}$ **do** ▷ $\forall n$
  $Calculate\_Loading(\lambda)$ ▷ e.g.Exhaustive/Iterative/Branch and Bound
  $distance = |P_{budget} - P(n)|$ ▷ Euclidean distance
  if $distance < min\_distance$ **then**
    $\lambda_{best} = \lambda$
    $\lambda_n = \lambda_n + \mu(P(n) - P_{budget})$
    $\mu = \mu * 2$
  else
    $\lambda = \lambda_{best}$
    $\mu = 1$
  end if
end while

2.9.6 Multi-User Greedy Spectrum Balancing

In [38], [3] and [39], Lee, Sonalkar and Cioffi presented a spectrum balancing algorithm known as the Multi-User Greedy Algorithm. It is based on a heuristic extension of the single user Levin-Campello Algorithm.

In the Levin-Campello Algorithm, bits are added to the channel which requires the least energy to do so, until the power budget of the line is reached. It is a conceptually simple algorithm, which produces the optimal discrete solution for the single user case. However, as noted in Section 2.9.2, when it is applied to the multi-user case in IWF, it produces sub-optimal results.

The Multi-User Greedy Algorithm attempts to extend the Levin-Campello algorithm to the multi-user case. The entire binder group is considered as a whole and bits are added incrementally to the channel and line with the lowest energy cost. The cost metric is defined as the power required to add one bit to a particular line/tone and the sum of the power increases required on all other lines on that tone in order to bring those lines back within the performance margin given the increased crosstalk. The objective of the multi-user greedy algorithm is to minimise the total power required to transmit a total rate-sum.

The cost function is calculated as shown in equation 2.35. $C(m,k)$ is the cost to add a bit to user m on tone k. $p(k)$ is a vector of the power spectral density on each line on tone k, for a given bit-vector $b(k)$. In equation 2.35 the vector e_m is a vector of zeroes with element m set to 1. This represents adding a bit to line m. Thus, the total power increase is calculated by the sum of the differences in the vector $p(k)$ before and after adding a bit to line m.

$$C(m,k) = \sum_{n=1}^{N} \left(\underset{b(k)+e_m}{p_n(k)} - \underset{b(k)}{p_n(k)} \right) \tag{2.35}$$

A simplified version of the Multi-User Greedy Algorithm is illustrated in Algorithm 8.

In [3] the authors proposed an algorithm to find arbitrary data rate points with the addition of a per line weight w_n and an algorithm to adjust the values of w_n on each iteration of the algorithm. Unfortunately, this algorithm is potentially

Algorithm 8 Multi-user Greedy Algorithm

repeat

$\text{argmin}_{m,k}\, C(m,k) = \sum_{n=1}^{N} \left(\underset{b(k)+e_m}{p_n(k)} - \underset{b(k)}{p_n(k)} \right)$

Increment $b_m(k)$

until All tones full

very slow or unstable as it relies on a arbitrarily chosen factor.

2.10 Conclusions

In this chapter, some popular DSM algorithms that have been developed in recent years were reviewed. It was shown that none of the current known algorithms are a perfect solution for implementing DSM. In particular, the level 2 DSM algorithms [17] and [34] discussed in this chapter are far too slow or intractable to be of practical use in realistic DSL deployments.

Level 2 DSM has two inherent advantages over level 1 techniques such as IWF. Firstly, as long as the algorithm is fast enough (i.e. less than the typical service time), the behaviour of users in the DSL network can be changed instantaneously, for example in response to changing data rate requirements or the addition of new users to the network. Secondly the performance margins can be set with exact precision in order to guarantee a consistent probability of error for each user in the network.

The remainder of this thesis investigates the development of new level 2 DSM algorithms with good data rate performance and lower complexity and execution times than currently known methods. In particular, the performance of the Multi-User Greedy Algorithm is investigated further in chapters 3 and 4 to form new algorithms which can produce solutions in a tractable time with good data rate performance.

In Chapter 3, a new level 2 DSM algorithm known as MIPB is developed and is shown to produce results comparable to the optimal solution given by OSB for a range of DSL scenarios, with a low enough complexity to be utilised on large bundle sizes (50-100 DSL lines). However, it currently has a limitation whereby

only one data rate point can be obtained, which is similar to that given by OSB with equal line weights.

In Chapter 4 another algorithm is developed which overcomes the limitation of MIPB and is able to calculate arbitrary points on the rate region. It is shown through simulation of various DSL network configurations that this algorithm produces rate regions that are practically as large as the optimal rate region given by OSB. Several rate search techniques for this algorithm are derived and it is shown that it can deliver a massive speed increase compared to ISB, up to 3850 times faster for a 7-User ADSL2+ downstream scenario.

Chapter 3

Multi-User Incremental Power Balancing for DSM

3.1 Introduction

This chapter investigates the design of an algorithm for multi-user spectrum optimisation in a Digital Subscriber Line (DSL) bundle. Over the last number of years, many algorithms have been proposed to tackle this problem [8] [35] [34] [17] [10]. As previously mentioned in Section 2.9 The task of optimising spectrum allocation for DSL bundles has come to be known as Dynamic Spectrum Management (DSM).

The optimal solution for the spectrum balancing problem is well known [17], where optimality is defined as the ability to achieve a combination of rates on the rate region boundary. All of these algorithms require some centralised control, implemented in a Spectrum Management Centre (SMC).

Spectacular performance gains are possible with DSM techniques over traditional Static Spectrum Managment (SSM). However, as discussed in Chapter 2, the problem with many of these techniques is that they are very slow or perhaps computationally intractable for practical scenarios. Furthermore, the convergence of many popular DSM algorithms has not been proven. In IWF, the performance margins will not always converge to the desired value, this is demonstrated later

in Section 3.4.3. Algorithms based on Lagrangian dual decomposition, such as OSB, ISB and SCALE, all exhibit convergence problems in some circumstances both in the power and rate constraints.

This chapter details the development of a new algorithm which delivers near optimal performance in a fraction of the time needed by previous methods and exhibits no convergence problems. The work presented here is developed from previously published research on greedy bit loading algorithms in [15] and [16].

All of the results presented in the rest of this thesis are generated from a custom built DSM simulator, the architecture of which is described briefly in Appendix B.1

3.2 Multi-User Greedy Bit-Loading Re-Examined

The Multi-User Greedy algorithm, described in Section 2.9.6 is a sub-optimal dynamic spectrum management algorithm. Its main advantages over other spectrum management algorithms are its tractability for large system sizes and it's non-iterative nature, which means there are no issues with convergence.

In this section, the performance of this algorithm will be examined and a new spectrum management algorithm will be derived from it with improved performance.

3.2.1 2-User Near Far Scenario

In order the examine the Multi-User Greedy Algorithms performance, the classic "near-far" scenario shown in Figure 2.6 will be initially chosen. To aid the analysis, the scenario will be evaluated with a bit-loading resolution of 0.01 bits per tone, rather than integer bits. This approximates a continuous bit distribution. It is particularly useful to approximate a continuous bit distribution for OSB, as this improves the convergence of the algorithm and enables full use of the power budgets on each line. With an integer distribution, OSB can converge towards a condition known as a "non-active power constraint" where one or more of the

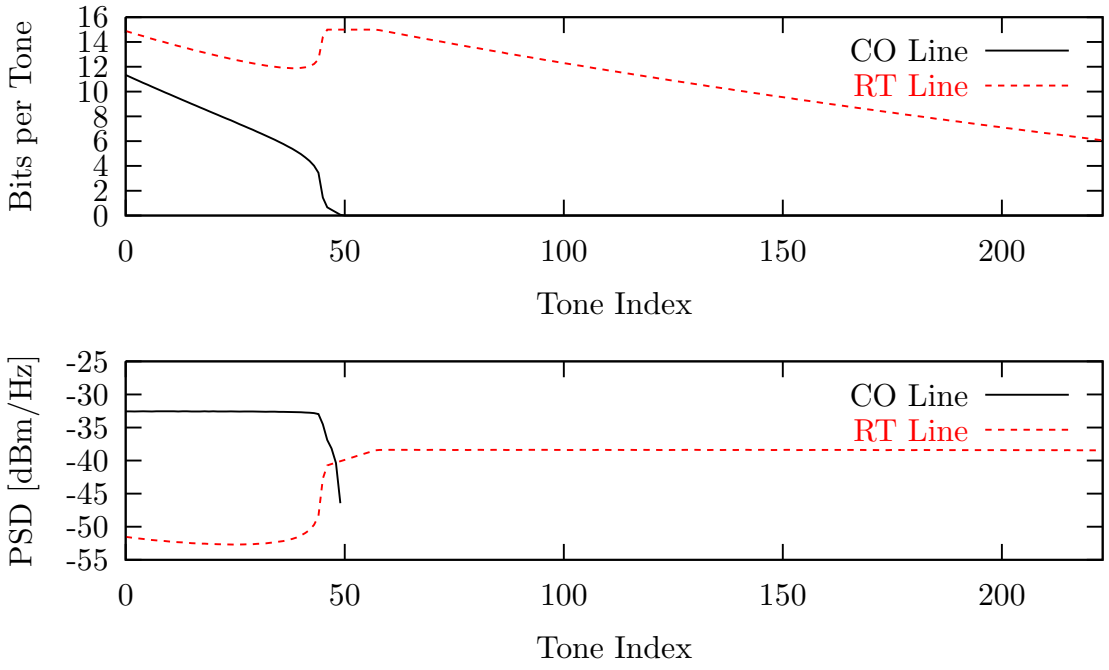


Figure 3.1: Optimal PSD and bit allocation for scenario in Figure 2.6 for downstream ADSL

Lagrangian multipliers converges to zero. In the case of a user's Lagrangian multiplier converging to zero, that user will not utilise their full power budget. It is found that some scenarios will not converge within arbitrary power constraints with OSB and ISB. Increasing the resolution of bit-loading mitigates this issue in the scenarios investigated in this thesis.

Figure 3.1 shows the optimal bit and PSD allocation for the scenario in Figure 2.6 for downstream ADSL with a 1% worst case FEXT crosstalk model and no alien crosstalk. The result was calculated using the OSB with Branch and Bound algorithm [37], with some adjustments to enable a non-integer bit allocation. To enable a non-integer bit allocation, each dimension in the region of possible solutions to the Lagrangian is further divided into fractions of bits. The results are summarised in Table 3.1.

To compare results with the optimal allocation, the PSD and bit allocation is calculated using the Multi-User Greedy bit loading algorithm [39] described in Section 2.9.6. The results given by the greedy algorithm are illustrated in Figure 3.2 and summarised in terms of data rate and transmit power in Table 3.2.

	CO Line	RT Line
Total Power (mW)	110.16	110.78
Date Rate (bits per frame)	357.69	2473.19

Table 3.1: Total transmit power and bit rates for optimal PSD and bit allocation for scenario in Figure 2.6

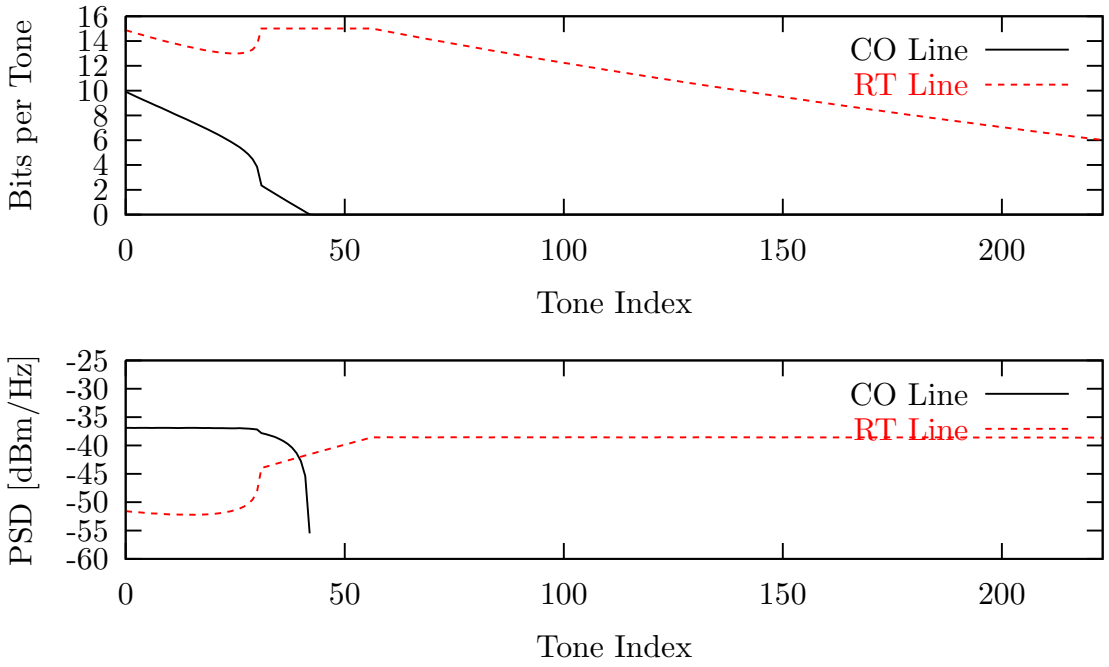


Figure 3.2: PSD and bit allocation for scenario in Figure 2.6 for downstream ADSL calculated with Multi-User Greedy bit loading algorithm

	CO Line	RT Line
Total Power (mW)	32.34	110.00
Date Rate (bits per frame)	242.47	2512.97

Table 3.2: Total transmit power and bit rates for PSD and bit allocation calculated with Multi-User Greedy algorithm for scenario in Figure 2.6

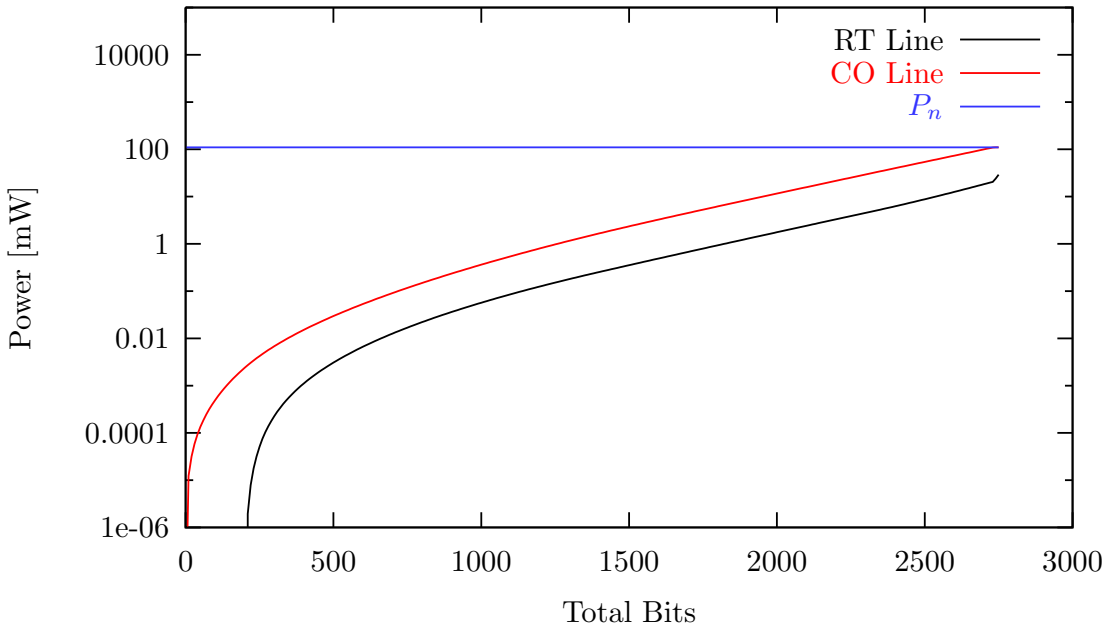


Figure 3.3: Graph of power used on each line against total bits loaded for Multi-User Greedy algorithm in scenario 2.6

3.2.1.1 Analysis of Results

By comparing Figure 3.1 and Figure 3.2, it can be seen that the PSD and bit allocations are a roughly similar shape. However, tables 3.1 and 3.2 show that the data rate of the CO line loaded with the greedy algorithm is 32.2% lower than the optimal solution. The data rate on the RT line is 1.6 % higher with the Multi-User Greedy compared to OSB. It is also noted that the power utilised by the CO line with the greedy method is much lower than it's power budget of 110mW at 32.32mW. This occurs because the CO line will not be allocated any more power as to do so would raise the transmit power of the RT line over its power budget due to increased crosstalk.

Figure 3.3 illustrates the power used on each line against the total number of bits loaded. From Figure 3.3, it can be seen that the RT line's power stays well ahead of the power used by the CO line throughout bit loading, until the RT line has utilised its entire power budget. At this point, a "deadlock" situation has occurred as the CO line cannot add any additional bits without breaking the RT line's power constraint even though it has power available in it's budget.

3.2.2 Multi-User Greedy Deadlock

It can be determined intuitively how this "deadlocked" situation arises. The Multi-User Greedy cost function is shown in equation 2.35. The cost is the sum of the power increases needed on all lines when a bit (or fraction of a bit in this analysis) is added on a particular line and tone. Since the RT line is much shorter than the CO line, the initial cost of bits on this line will be much lower than the CO line. Although the cost function in equation 2.35 includes extra power introduced due to crosstalk, these terms will be initially zero when the bit loading begins.

Thus, the RT lines total power and bits initially increase rapidly until the the minimum cost to add one more bit is on the CO line. This occurs due to the exponential bit/power relationship in equation 2.18. As a result of the RT loading a higher number of bits at low frequencies, the CO line cannot add many bits on these frequencies without causing the RT line to exceed its own power budget. This results in the "deadlock" scenario, where the CO line is not given an opportunity to utilise its full power budget.

3.2.3 Power Penalty Function

In a first attempt to modify the Multi-User Greedy algorithm in order to improve performance, a new cost function will be developed. To this cost function a new term is added, which is referred to as the "power penalty function" $wp(n)$. The goal of this function is to ensure that all lines get an opportunity to use their entire power budget. It is posited that this will result in a fairer PSD and bit allocation than with the Multi-User Greedy algorithm. The value of $wp(n)$ is a function of the total power currently used by line n and the average power used by all lines.

The premise is that the value of $wp(n)$ should be a value larger than 1 for a line that has used more power than the average and less than 1 for a line which has used less than the average. The intention is to keep the lines used-power similar as they load bits so that they reach their power budget in synchronisation with

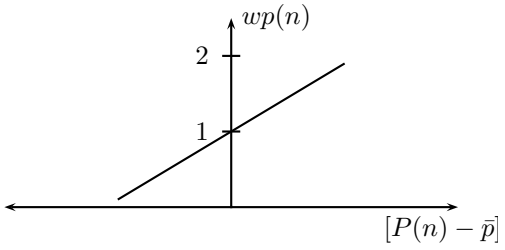


Figure 3.4: A linear $wp(n)$ function

each other, rather than exhibiting the "deadlock" effect noted previously with the Multi-User Greedy algorithm. An example of a valid linear $wp(n)$ function is shown in Figure 3.4.

The initial experiment is undertaken with the penalty function applied only to the energy cost of the bit being added, rather than including the increased power due to crosstalk in the cost function as in the original Multi-User Greedy algorithm. The new cost function is shown along with a linear $wp(n)$ function in Algorithm 9 which represents a simplified version of the bit loading algorithm itself. The gradient of the $wp(n)$ is chosen arbitrarily at this stage.

Algorithm 9 Greedy Loading Algorithm with linear power penalty function and zero crosstalk penalty

repeat

$\text{argmin}_{n,k}\, wp(n) \times \left({}_{b(k)+e_n}p_n(k) \; - \; {}_{b(k)}p_n(k) \right)$

Increment $b_n(k)$

until All tones full

function $wp(n)$

return $1e3 \times (P(n) - \bar{p}) + 1$

end function

The results of bit loading using Algorithm 9 is shown are shown in Figure 3.5 and Table 3.3.

From Table 3.3, it can be seen that each line did indeed utilise their entire power budget as expected. Figure 3.6 illustrates the amount of power used by each line against the total number of bits loaded. It can be seen that the power usage of each line converges as they approach their power budget. However,

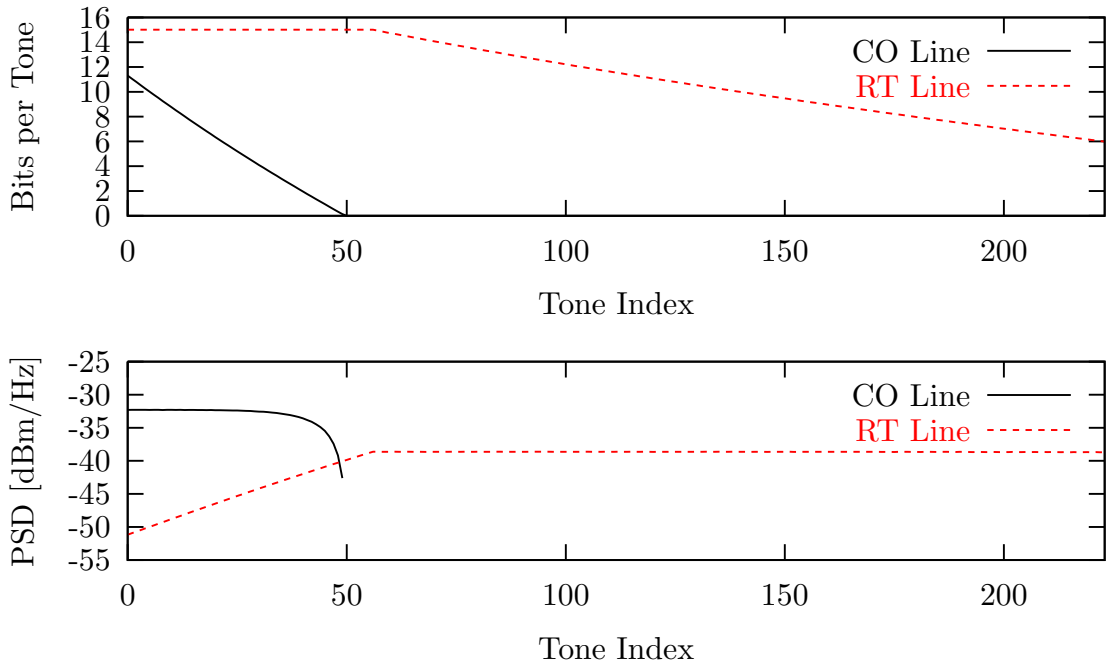


Figure 3.5: PSD and bit allocation for scenario in Figure 2.6 for downstream ADSL calculated with Algorithm 9

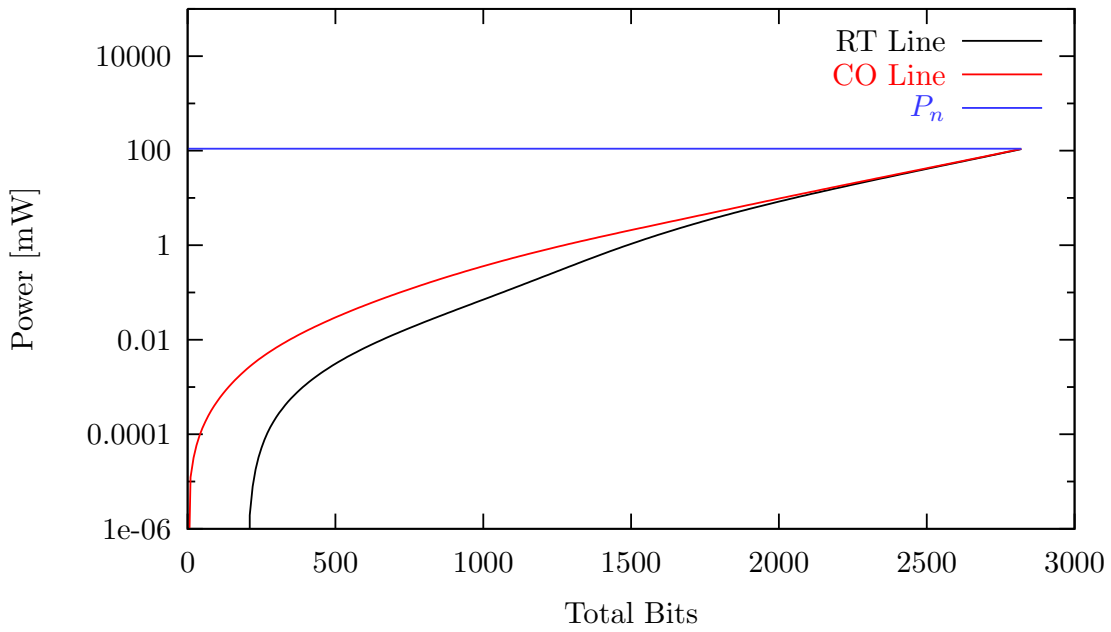


Figure 3.6: Graph of power used on each line against total bits calculated with Algorithm 9 in scenario 2.6

	CO Line	RT Line
Total Power (mW)	109.99	109.99
Date Rate (bits per frame)	273.05	2550.13

Table 3.3: Total transmit power and bit rates for PSD and bit allocation calculated with Algorithm 9 for scenario in Figure 2.6

there are marked differences in the spectrum allocation between Figure 3.5 and the optimal spectrum allocation in Figure 3.1.

The key difference between the optimal spectrum allocation and the spectrum in Figure 3.5 are in the low frequency range, between tone index 0 and 50. In the optimal allocation, the RT line "yields" to the longer CO line in the low frequencies. As the crosstalk coupling is very strong at these frequencies from the RT line into the CO line, this yielding enables a significant increase in the data rate of the CO line. By sacrificing a small percentage of the RT line's data rate, a much larger increase in the rate of the CO line is possible, resulting in a larger total capacity in the bundle.

Having made this observation of the performance degradation of Algorithm 9, the next logical step is to introduce crosstalk into the cost function. The resulting algorithm is shown in Algorithm 10.

Algorithm 10 Greedy Loading Algorithm with linear power penalty function

repeat

$$\text{argmin}_{m,k} \sum_{n=1}^{N} wp(n) \times \left(\underset{b(k)+e_m}{p_n(k)} - \underset{b(k)}{p_n(k)} \right)$$

Increment $b_m(k)$

until All tones full

function $wp(n)$

return $1e3 \times (P(n) - \bar{p}) + 1$

end function

The results generated by Algorithm 10 are shown in Figure 3.7, Figure 3.8 and Table 3.4 and Figure 3.8. By comparing these with the results for the optimal solution in Figure 3.1 and Table 3.1, it can be seen that the generated solutions

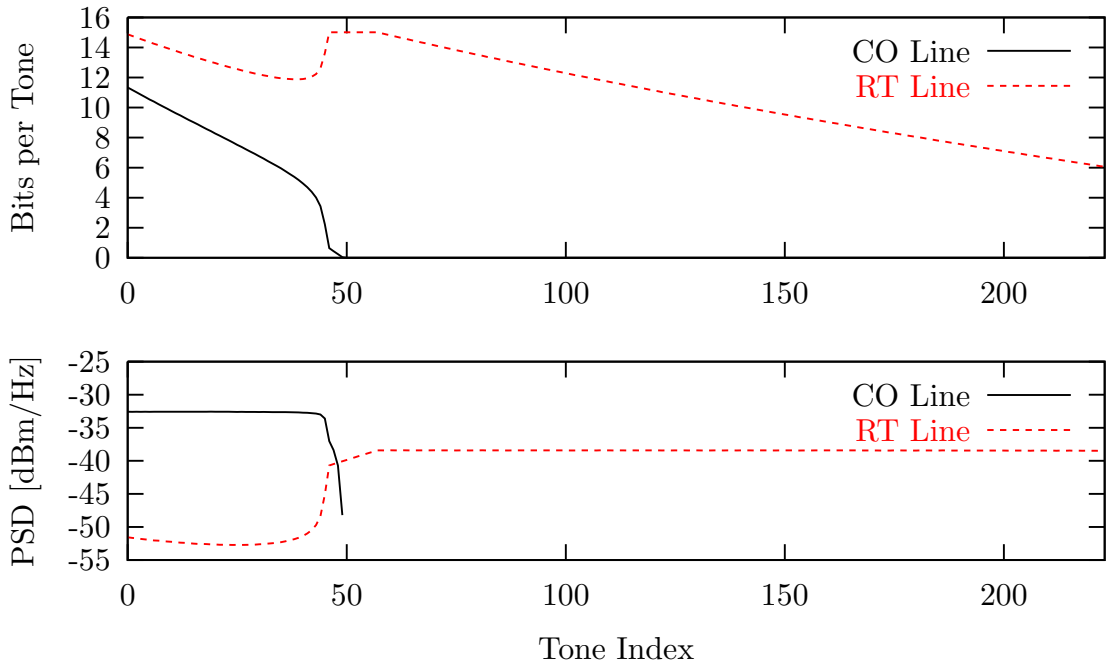


Figure 3.7: PSD and bit allocation for scenario in Figure 2.6 for downstream ADSL calculated with Algorithm 10

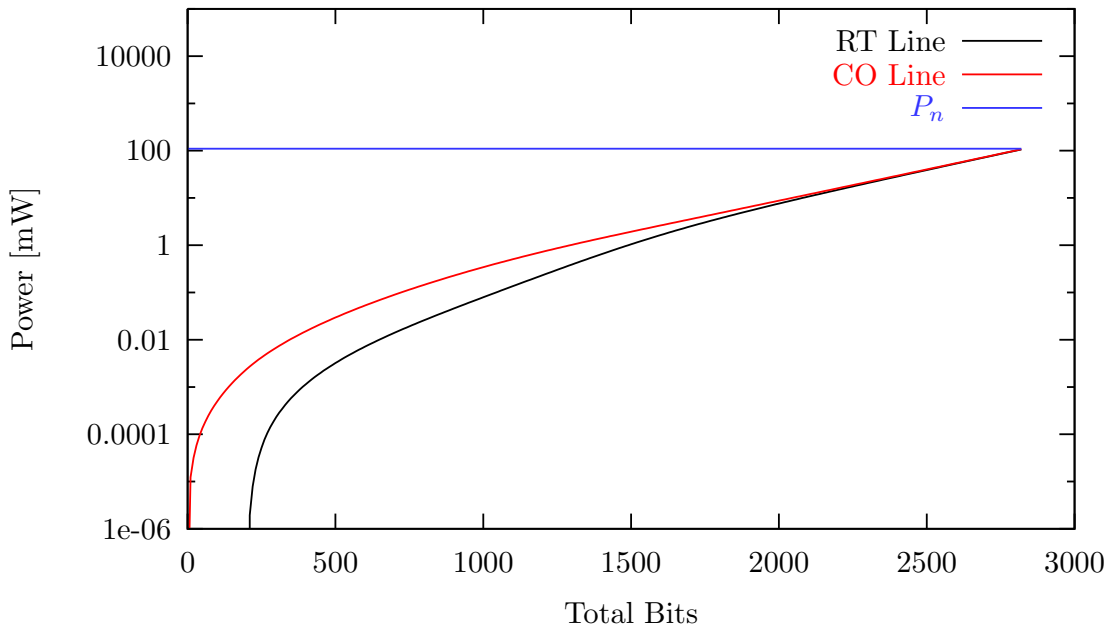


Figure 3.8: Graph of power used on each line against total bits calculated with Algorithm 10 in scenario 2.6

	CO Line	RT Line
Total Power (mW)	109.99	109.99
Date Rate (bits per frame)	358.42	2470.55

Table 3.4: Total transmit power and bit rates for PSD and bit allocation calculated with Algorithm 10 for scenario in Figure 2.6

are very similar in terms of spectrum shape and also in data rates.

The graph of power used against bits loaded in Figure 3.8 shows that the original objective of keeping used power synchronised during bit loading is achieved.

3.2.4 Adaptive Power Penalty Function

In algorithms 9 and 10, a linear $wp(n)$ function was used with an arbitrarily chosen gradient. In fact the value was deduced from experiments so that it would not produce a negative value of $wp(n)$ whilst still being large enough to equalise the total transmit power of each user.

Unfortunately, it seems that an arbitrarily chosen large number does not work very well in general, so a more general function must be created in order to balance the used power of each line. The term general in this case is used to mean a function that does not require any experimentally derived values.

By observing Figure 3.7 it can be seen that costs per bit (or fraction of) have a large range of values. For the "near-far" scenario in Figure 2.6 with a resolution of 0.01 bits, the cost (a weighted sum of power in mW) ranges from about $1e-10$ to $1e-4$. The costs increase exponentially as the number of bits loaded increases.

It was noted experimentally that if $wp(n)$ is set equal to one for negative values of $P(n) - \bar{p}$ as in Figure 3.9, virtually identical results for the scenarios already investigated are obtained. However, in situations where one line cannot reach its own power constraint (due to bit cap), smoother spectra are produced. Another advatange of setting the value to $wp(n)$ to one for negative values of $P(n) - \bar{p}$ is that there is no possibility of obtaining a negative value for $wp(n)$,

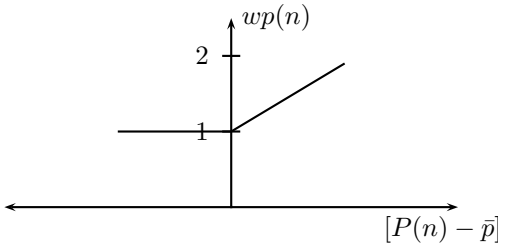


Figure 3.9: A $wp(n)$ function with $wp(n) = 1$ for negative values of $P(n) - \bar{p}$

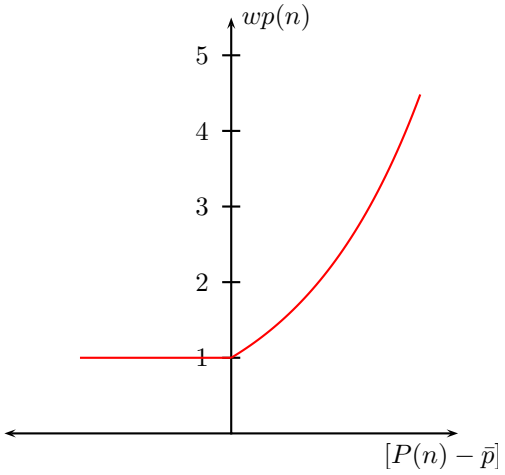


Figure 3.10: A $wp(n)$ function which is exponential for positive values of $P(n) - \bar{p}$ and $wp(n) = 1$ for negative values of $P(n) - \bar{p}$

as there would be a possibility with a linear function of $P(n) - \bar{p}$. A negative value of $wp(n)$ would produce a negative cost value for a particular bit, which has no physical meaning and would cause undefined behaviour in the bit loading algorithm.

The updated $wp(n)$ function was chosen to be an exponential function for positive values of $P(n) - \bar{p}$. In addition, the value of $P(n) - \bar{p}$ is normalised before the exponential function is applied. This normalisation is carried out in order to remove the need for an experimentally deduced constant to determine the steepness of this function. Each value of $P(n) - \bar{p}$ is normalised by the value of the total power cost of the last bit added. The shape of the new $wp(n)$ function is illustrated in Figure 3.10.

Algorithm 11 Greedy Loading Algorithm with adaptive power penalty function

repeat

$$\text{argmin}_{m,k} \sum_{n=1}^{N} wp(n) \times \left(p_{n}(k)_{b(k)+e_m} - p_{n}(k)_{b(k)} \right)$$

Increment $b_m(k)$

until All tones full

function $wp(n)$

return $e^{\frac{P(n)-\bar{p}}{\Delta p_{last}}}$

end function

	CO Line	RT Line
Total Power (mW)	109.99	109.99
Date Rate (bits per frame)	349.24	2479.52

Table 3.5: Total transmit power and bit rates for PSD and bit allocation calculated with Algorithm 11 for scenario in Figure 2.6

The results given by Algorithm 11 are shown in Figure 3.11, Table 3.5 and Figure 3.12. It is clear that the spectrum shape and data rates are very similar to the optimal method calculated by OSB. The graph of power used against bits loaded in Figure 3.12 shows that the power achieves synchronisation earlier than with the static constant value, without the need for experimentally determining the value of the gradient.

3.3 Multi-User Incremental Power Balancing

The algorithm listed in Algorithm 11 will hereafter be referred to as Multi-User Incremental Power Balancing (MIPB). It is referred to as MIPB as it attempts to balance or equalise the transmit powers of multiple DSL users incrementally during the bit loading process. In this section, the performance of MIPB is examined and compared with other DSM algorithms.

Algorithm 11 shows a coarse grained representation of the calculations required to execute the MIPB algorithm. However, implementing this as written would result in a vast number of redundant calculations of $wp(n)$ and the calcu-

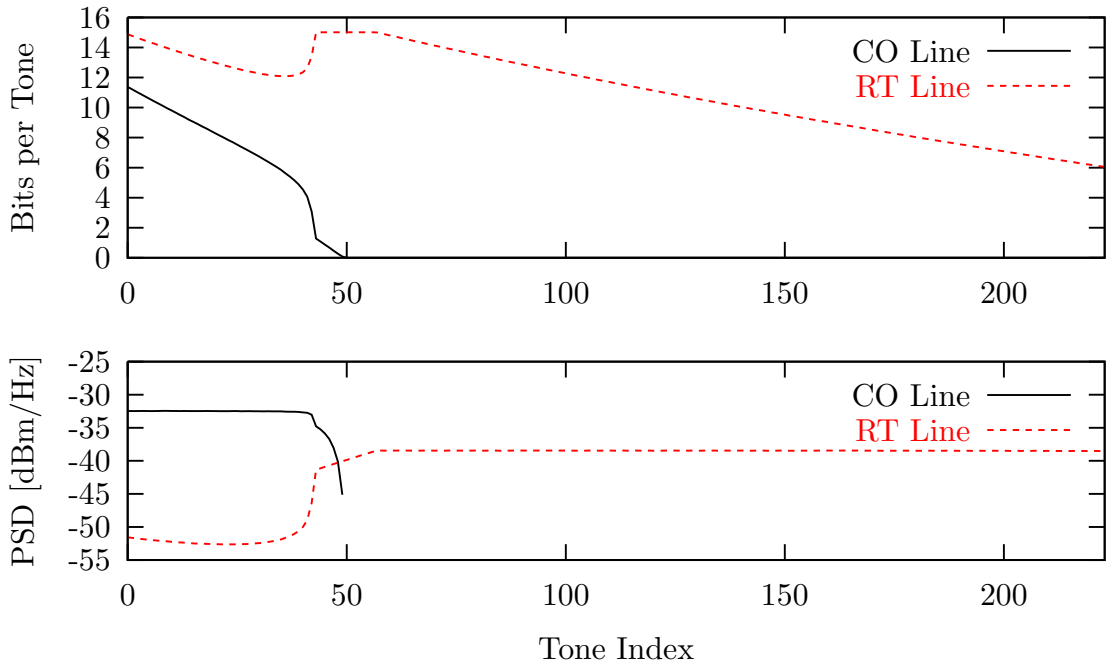


Figure 3.11: PSD and bit allocation for scenario in Figure 2.6 for downstream ADSL calculated with Algorithm 11

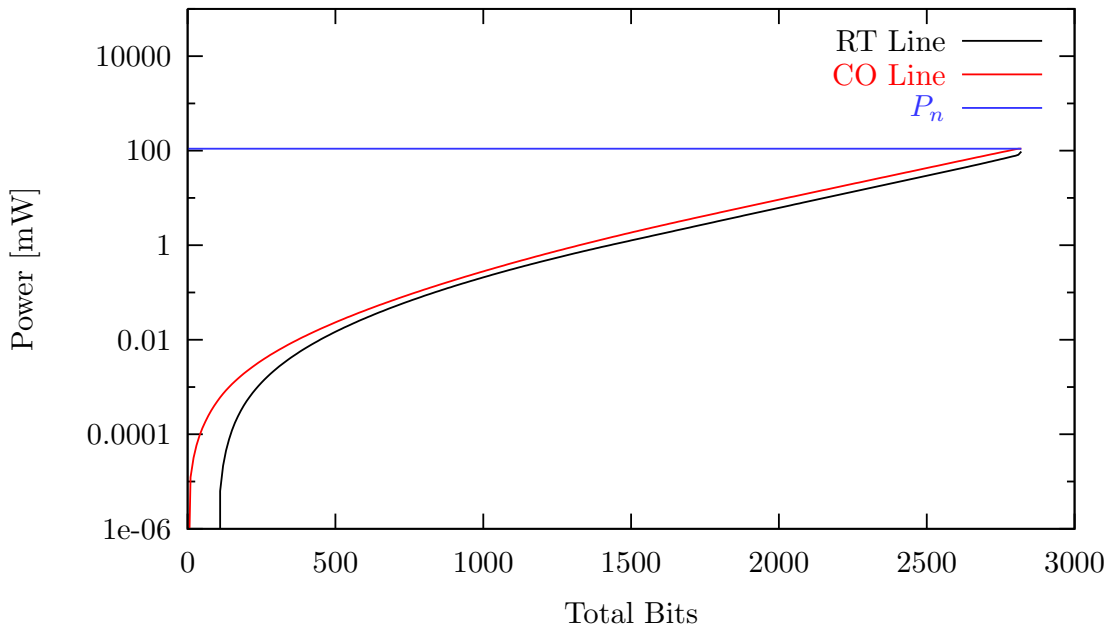


Figure 3.12: Graph of power used on each line against total bits calculated with Algorithm 11 in scenario 2.6

Algorithm 12 Multi-User Incremental Power Balancing (MIPB)

repeat
  $\text{argmin}_{n,k}\, cost\_matrix[n][k]$
  Increment $b_n(k)$
  for $n = 1 \ldots N$ **do**
    $\Delta p[n][k] = calc\_delta\_p(n, k)$ ▷ Δp update on tone k
  end for
  for $n = 1 \ldots N$ **do** ▷ Cost matrix updates
    for $k = 1 \ldots K$ **do**

$$cost\_matrix[n][k] = \sum_{n=1}^{N} wp(n) \times \Delta p[n][k]$$

    end for
  end for
until All tones full
function $wp(n)$
  return $e^{\frac{P(n)-\bar{p}}{\Delta p_{last}}}$
end function

lations of energy costs to add one bit on each line/tone.

In a realistic implementation, the current values of the incremental energy costs are stored in a Δp matrix and only updated when required. The cost matrix unfortunately has to be updated on each bit added, as it is a sum of products of the Δp values and $wp(n)$.

Algorithm 12 illustrates the algorithm in slightly more detail, showing both the Δp update step and the cost matrix recalculation.

3.4 Simulation Results

In this section, simulation results will be presented for a variety of different DSL network scenarios to compare the performance of MIPB against other prominent DSM algorithms.

3.4.1 2 User Near-Far ADSL Downstream Scenario

The 2 User Near-Far scenario is illustrated in Figure 2.6. The network termination point for the near user is in a fibre-fed remote terminal located 3km from the central office. The far user's network termination point is located in the central

Band Plan	ADSL Downstream
Power Budget	20.4dBm
Uncoded Gap	9.95dB
Line Gauge	AWG 24
Xtalk Model	1% Worst Case
PSD Mask	None

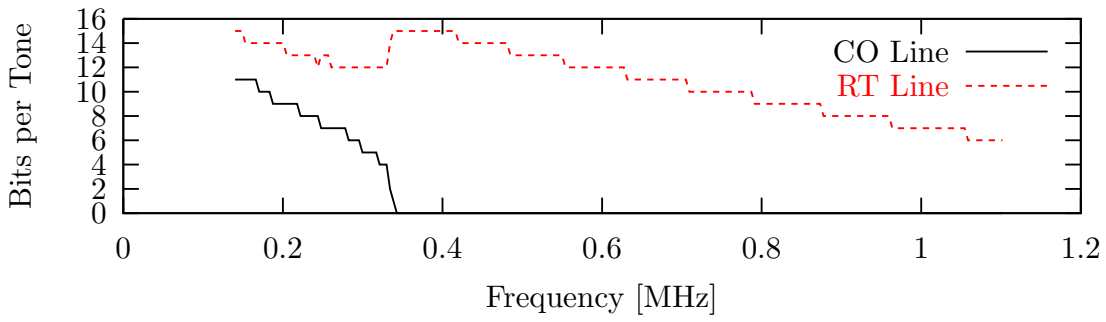
Table 3.6: Simulation Parameters for the 2 User Near-Far scenario in Figure 2.6

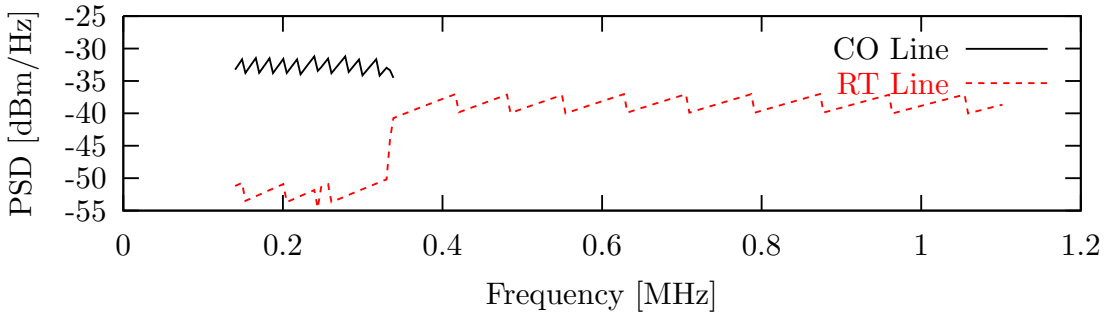
office. As previously discussed in Section 2.8 this is a particularly poor scenario for traditional waterfilling algorithms and is often used as a benchmark for DSM algorithms [40] [41].

The parameters for this scenario are shown in Table 3.6. By visual inspection of Figure 3.13, the bit and spectrum allocations produced by MIPB and OSB are very similar. Both spectra show a power backoff from the RT line to approximately -52.5 dBm/Hz in the low frequency range up to approximately 320kHz.

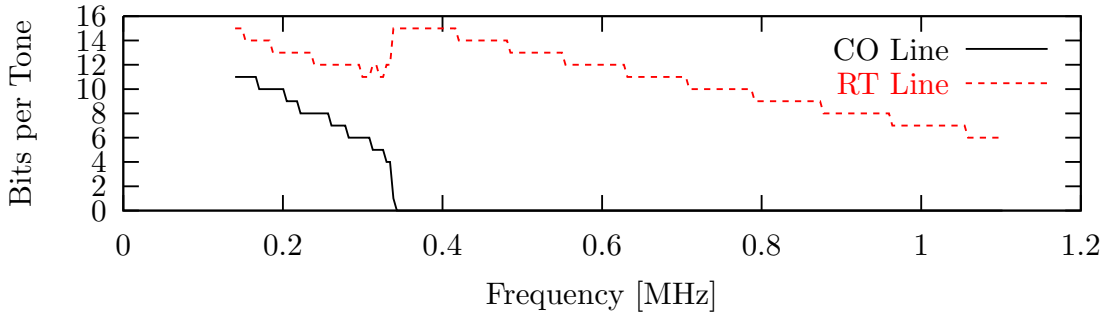
Table 3.7 shows the total output power and data rates on each line for the two algorithms. The total data rate in the bundle is identical for each algorithm, with a very small difference in the individual rates equal to 60kb/s.

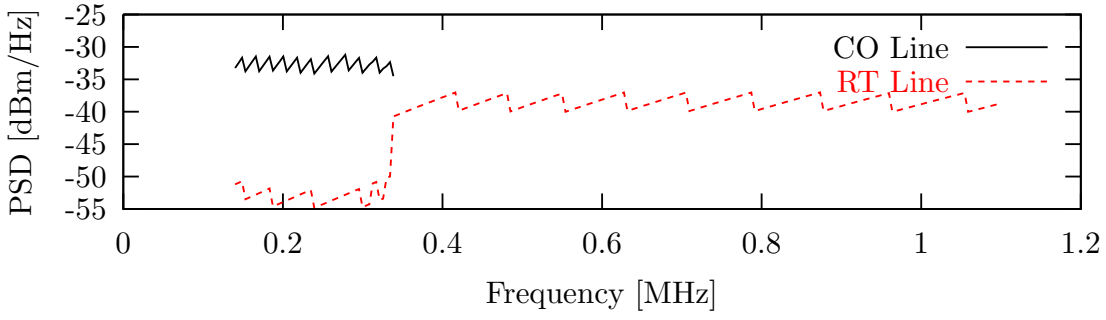
Figure 3.14 shows the PSD allocated to each line using OSB and MIPB on the same axis. On this figure, it is clear that the power allocations are virtually identical, expect for a few tones.





(a) OSB





(b) MIPB

Figure 3.13: Bit and spectrum allocation given by OSB and MIPB for the scenario in Figure 2.6

	CO Line	RT Line
Total Power (mW)	109.67	109.81
Date Rate (bits per frame)	371	2451

(a) MIPB

	CO Line	RT Line
Total Power (mW)	109.86	110.16
Date Rate (bits per frame)	356	2466

(b) OSB

Table 3.7: Total transmit power and bit rates for PSD and bit allocation calculated with MIPB and OSB for scenario in Figure 2.6

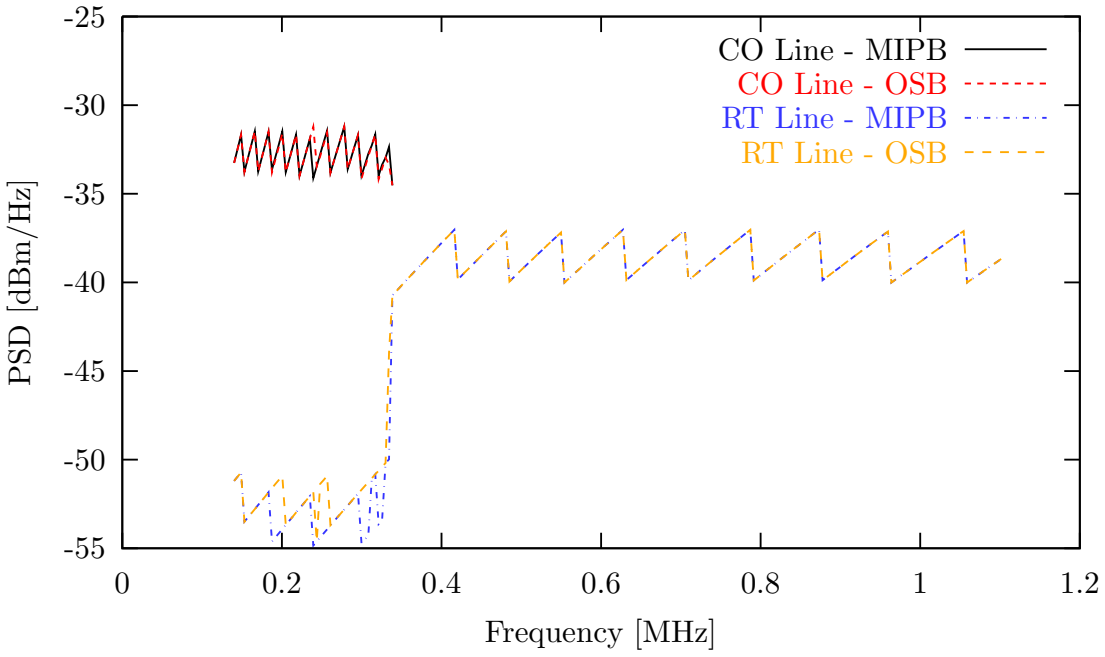


Figure 3.14: PSD allocation produced by OSB and MIPB for the scenario in Figure 2.6 shown on the same axis

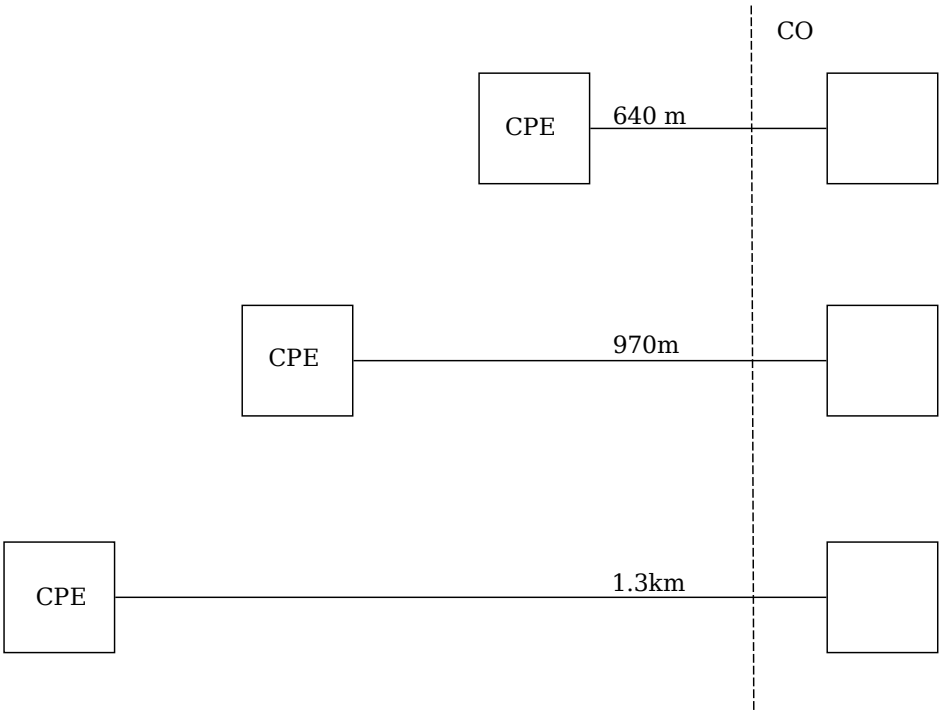


Figure 3.15: A 3-User Upstream VDSL2 scenario

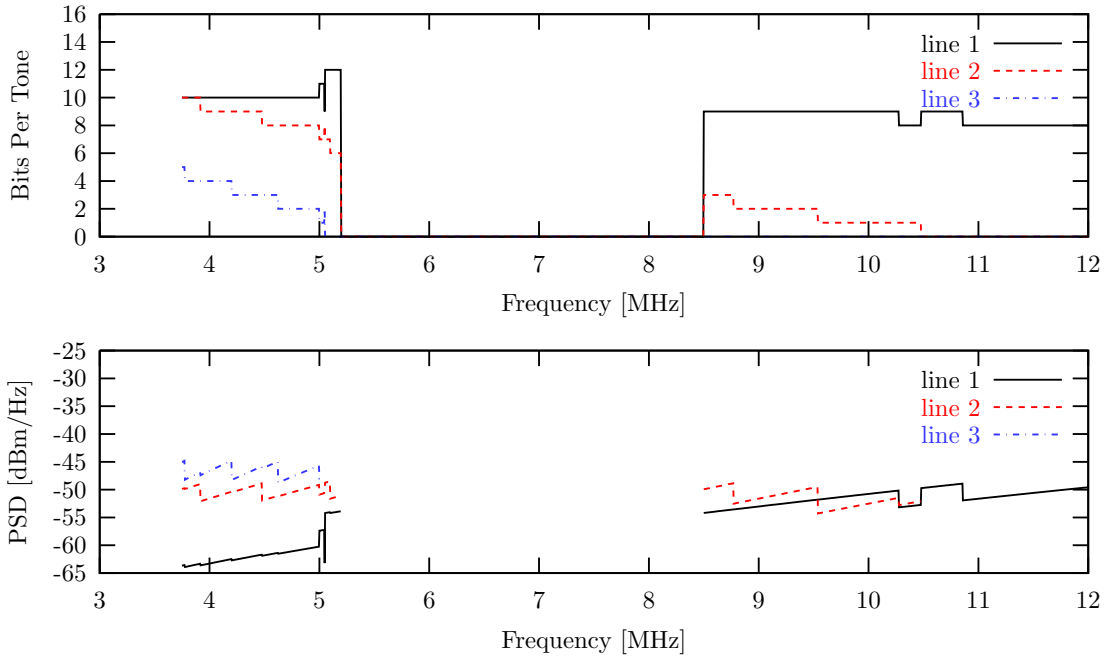
3.4.2 3-User VDSL2 Upstream

The next scenario examined is a 3-User VDSL2 upstream scenario. The NT point for each modem is in the central office and the length of each line is 640m, 970m and 1.3km respectively. This configuration is illustrated in Figure 3.15.

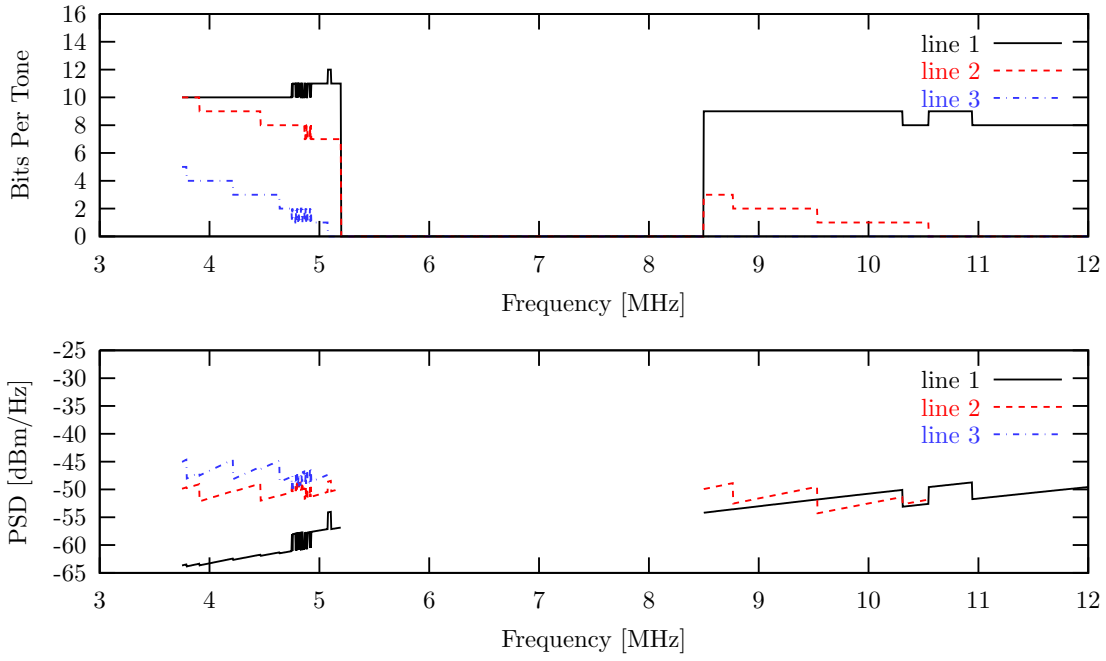
The band plan used is that of VDSL2 17a [42] [43] in which there are two upstream bands used. The first band used is between 3.75MHz and 5.2MHz whilst the second begins at 8.5MHz and stops at 12MHz. The simulation parameters used for this scenario are summarised in Table 3.8.

In this scenario, the beta model adjustment [23] is utilised using line indices 8,16 and 21 from [25].

By observation of the bit and spectrum allocations of OSB and MIPB for this scenario in Figure 3.16, the general similarities between the OSB allocation and that of MIPB are evident. In Table 3.9, the resulting data rates for each are presented. The individual rates are roughly equivalent, however, the average rate



(a) OSB



(b) MIPB

Figure 3.16: Bit and spectrum allocation for the VDSL2 scenario in Figure 3.15 calculated using OSB with equal line weights and MIPB

Band Plan	VDSL2 17a Upstream
Power Budget	14.5dBm
Uncoded Gap	9.95dB
Line Gauge	AWG 24
Xtalk Model	1% Worst Case + Beta Model
PSD Mask	None

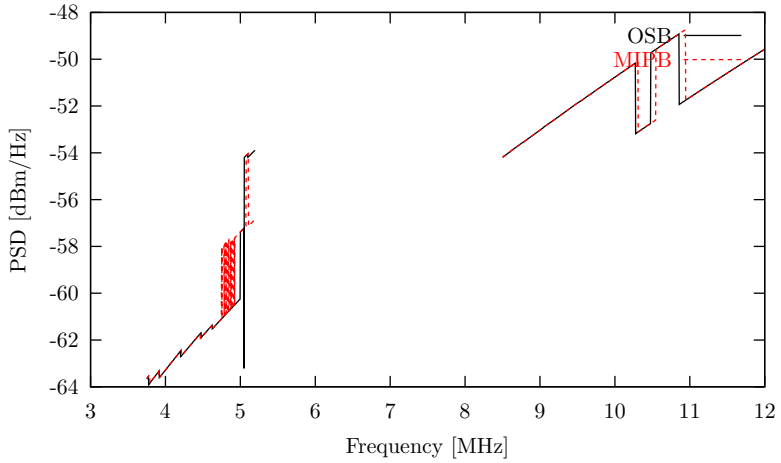
Table 3.8: Simulation parameters for 3-User VDSL2 upstream scenario

	OSB	MIPB
Line 1	41.74Mbps	41.86Mbps
Line 2	14.36Mbps	14.38Mbps
Line 3	3.63Mbps	3.516Mbps

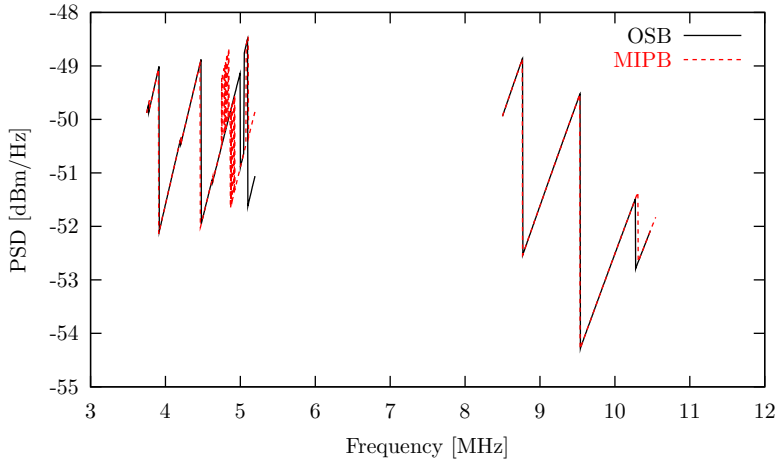
Table 3.9: Data rates on all lines for scenario in Figure 3.15 with OSB and MIPB

given by MIPB is slightly higher at 20.28Mbps, as opposed to 20.23 Mbps given by OSB.

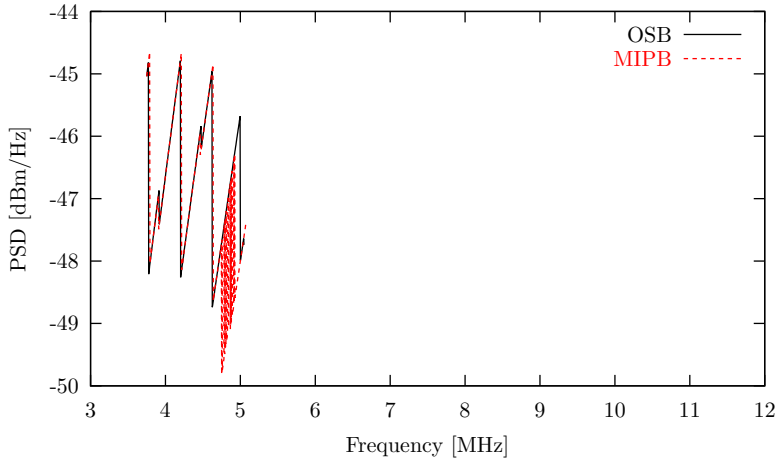
Figure 3.17 shows the PSD given by each algorithm on each line in Figure 3.15. Once again, it is clear that MIPB gives a spectrum allocation very close to that of the OSB for this particular scenario.



(a) Line 1



(b) Line 2



(c) Line 3

Figure 3.17: Comparison of PSDs produced by equal line weighted OSB and MIPB for the upstream VDSL2 scenario in Figure 3.15

Band Plan	ADSL2+ Downstream
Power Budget	20.4dBm
Uncoded Gap	9.95dB
Target Performance Margin	3dB
Line Gauge	AWG 24
Xtalk Model	1% Worst Case + Beta Model
PSD Mask	None

Table 3.10: 6 User ADSL2+ downstream simulation parameters

3.4.3 6-User ADSL 2+ Downstream

In this section, MIPB will be tested against various DSM algorithms for a 6 user ADSL2+ downstream scenario. In this deployment, there are 6 lines of varying length with two remote terminals located at 500m and 1km from the central office. This is illustrated in Figure 3.18. The results will be simulated based on both the 1% worst case FEXT model and the 1% worst case model plus the beta model adjustment. The OSB, ISB and IWF algorithms will be used for comparison against MIPB. The IWF algorithm in each simulation used is 20 iterations of the LC-RA algorithm.

3.4.3.1 Beta FEXT Model

Figure 3.19 shows the bit and spectrum allocation generated by OSB and MIPB for this 6 user scenario with the beta FEXT model introduced in Section 2.4.2 and the simulation parameters in Table 3.10. Line indices 0-6 are used from [25].

Table 3.11 shows the data rates for this scenario given by OSB, ISB, IWF and MIPB. It is noted that the results are quite similar for all algorithms except IWF, which shows a small drop in the average data rate and that of the maximum data rate in the bundle.

As OSB, ISB and MIPB are all level-2 DSM algorithms, the PSDs are calculated centrally with knowledge of the cross talk coupling functions. This guaran-

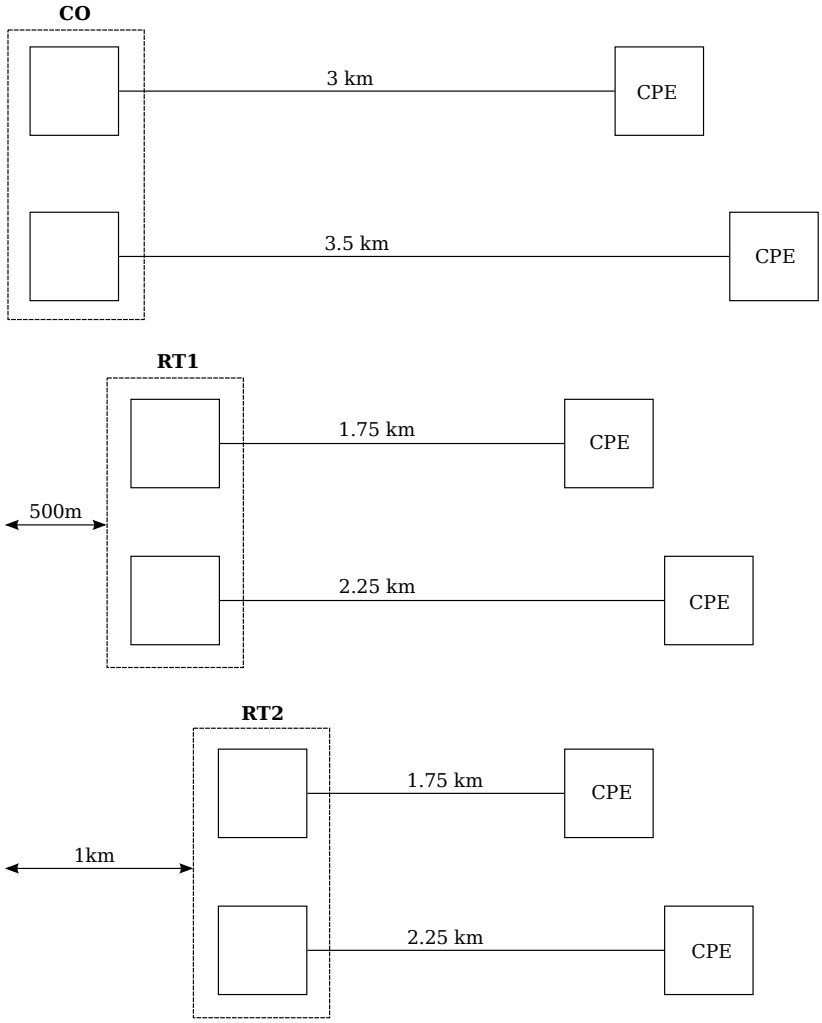
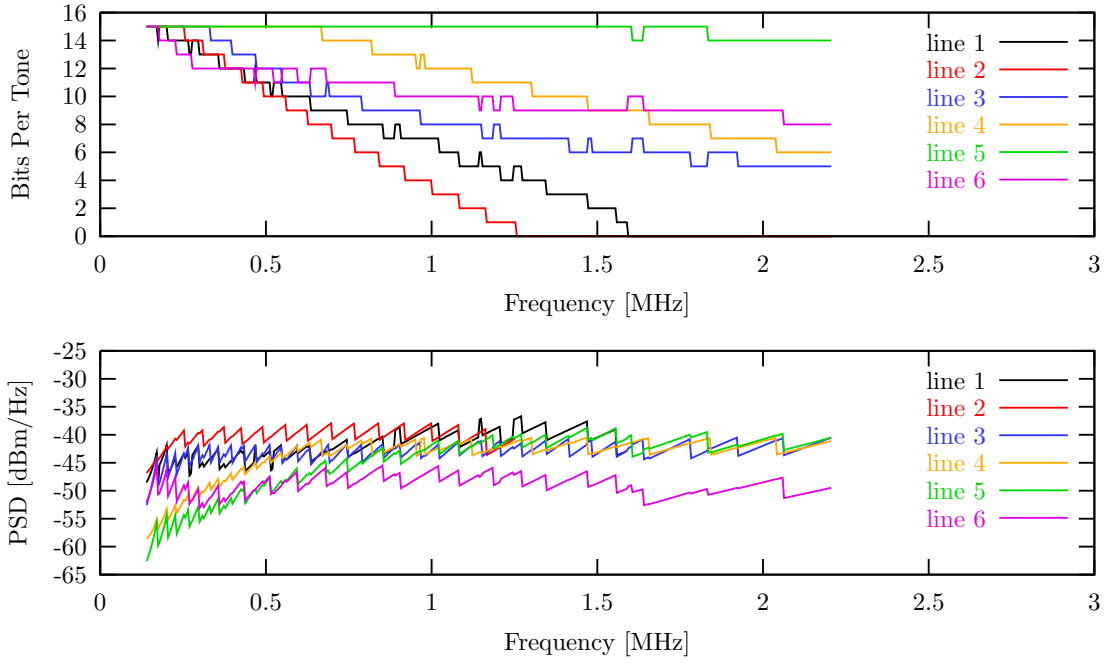
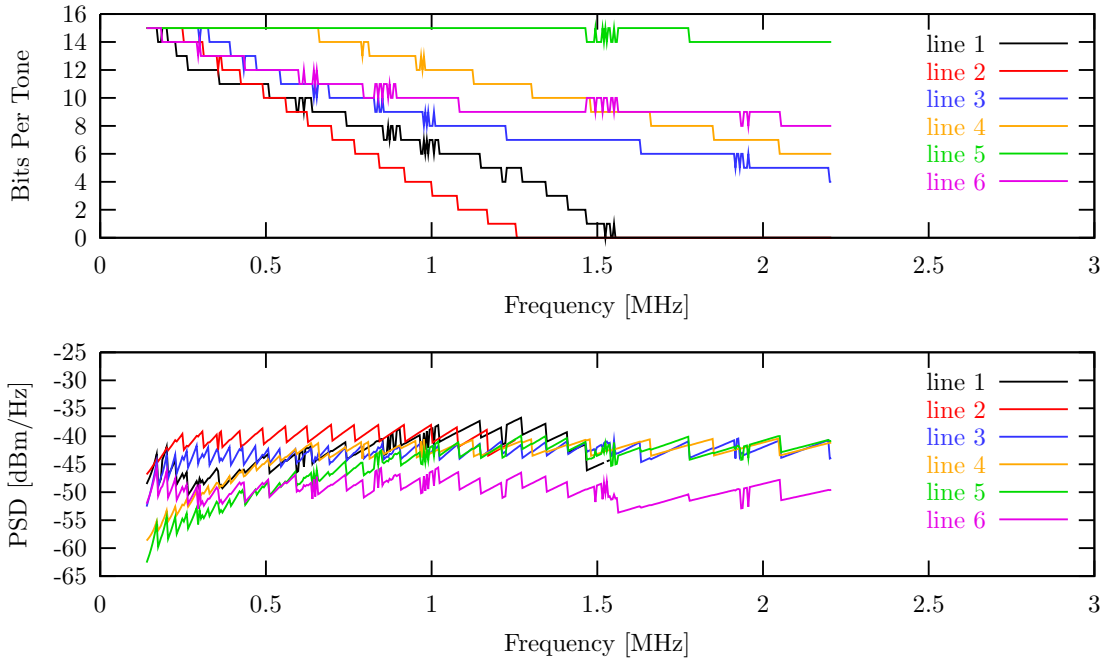


Figure 3.18: 6 User ADSL2+ downstream scenario with two remote terminals



(a) OSB



(b) MIPB

Figure 3.19: Bit and PSD allocation for the 6 line ADSL 2+ scenario is Figure 3.18 with parameters in Table 3.10

	OSB	ISB	IWF	MIPB
Average Rate (Mbps)	17.38	17.30	17.11	17.37
Minimum Rate (Mbps)	8.07	8.05	8.03	8.05
Maximum Rate (Mbps)	28.42	28.51	25.94	28.34

Table 3.11: Achievable data rates for scenario in Figure 3.18 given parameters in Table 3.10

tees that the performace margins on each tone are set at their desired level, 3dB in this case. However, IWF does not guarantee that. Although there is a certain level of convergence, there are still many tones which are not at their desired margin. Bit/Gain-swapping techniques [44] will bring those out-of-margin tones back to their desired level with some loss of performance. The performance margins on all lines after IWF for the scenario are illustrated in Figure 3.20. In the worst case, some tones on line 5 have a margin of approximately 0.25dB making them significantly more susceptible to impulse noise.

3.4.3.2 1% Worst Case FEXT Model

In this section, the same 6 user scenario will be investigated using the 1% worst case FEXT model. Figure 3.21 shows the bit and PSD allocation for both OSB and MIPB and it is observed they are broadly similar, except for some noise in the bit allocations for MIPB.

The resulting data rates for this scenario generated by OSB, ISB, IWF and MIPB are shown in Table 3.12. Compared to the results with the beta FEXT model adjustment, ISB and IWF fare considerably worse in this case whilst MIPB exhibits performance very close to that of OSB. IWF also exhibits poorer convergence of performance margin with the higher FEXT given by the 1% model. The margins on all lines are shown in Figure 3.22. It is concluded from comparing figures 3.20 and 3.22 that the convergence of the performance margin in IWF becomes poorer as the crosstalk coupling increases. Once again, this results in a

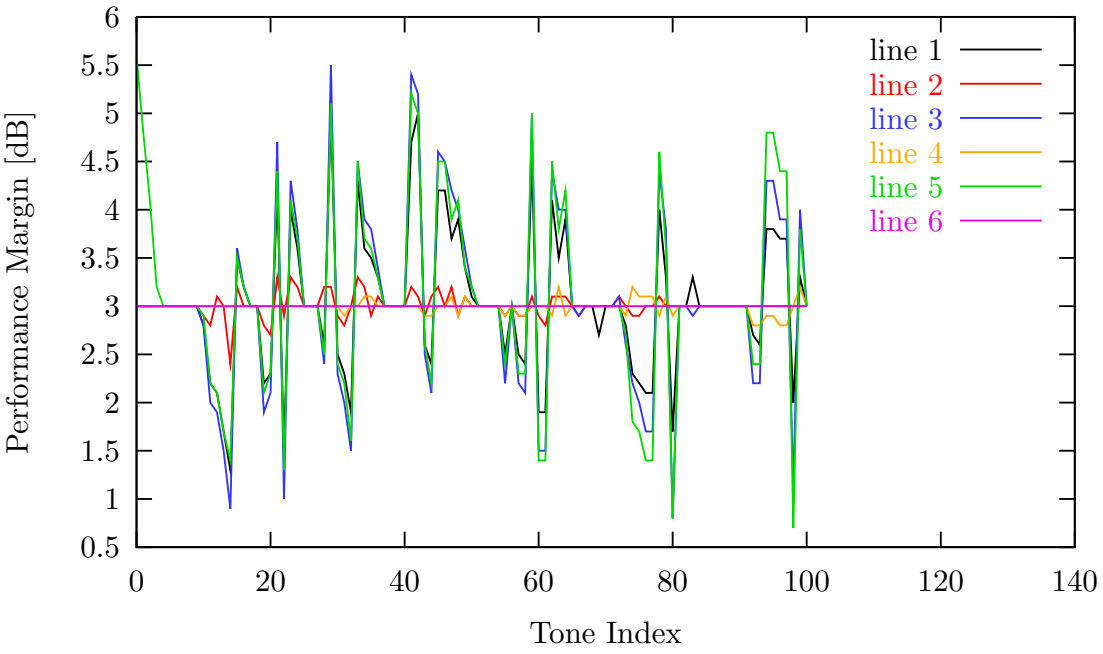
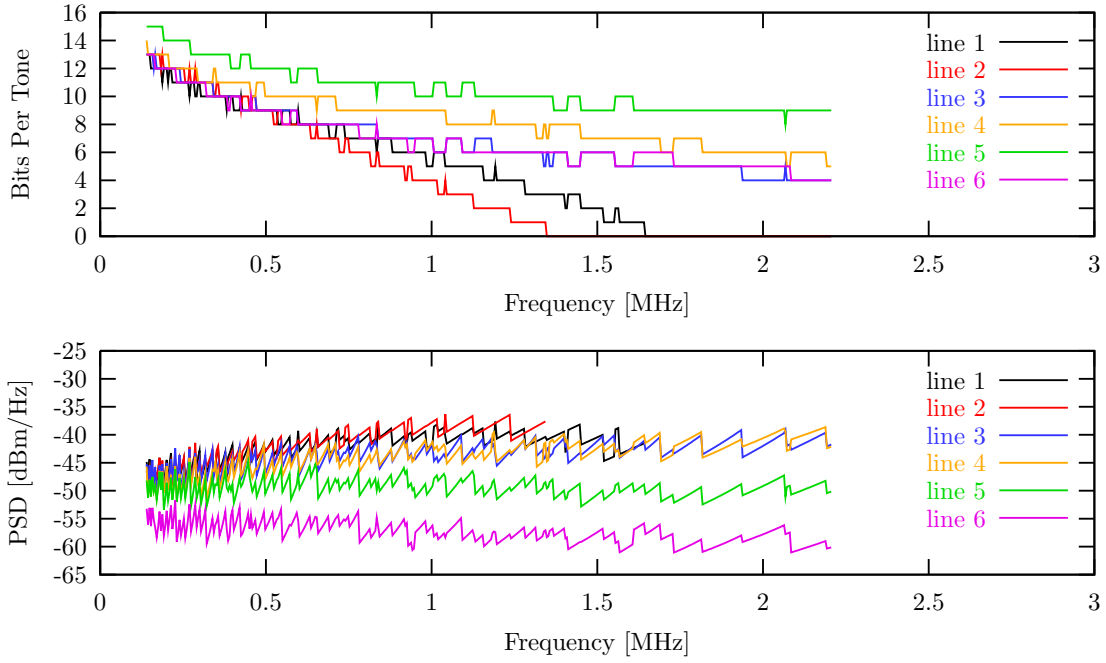


Figure 3.20: Performance margins on the first 100 tones for all lines from the network in Figure 3.18 using the Iterative Waterfilling Algorithm

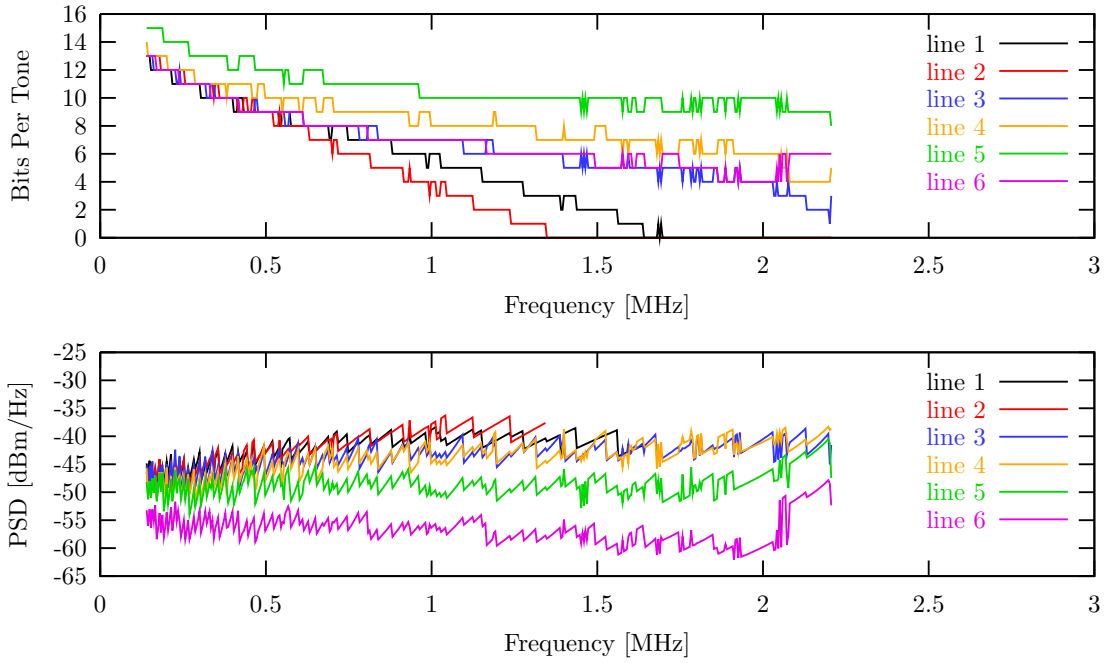
lower tolerance to impulsive noise, save for significant gain/bit-swapping.

	OSB	ISB	IWF	MIPB
Average Rate (Mbps)	13.14	12.79	11.88	13.08
Minimum Rate (Mbps)	7.28	7.17	7.27	7.20
Maximum Rate (Mbps)	20.22	17.4	20.06	20.54

Table 3.12: Achievable data rates for scenario in Figure 3.18 using the 1% worst case FEXT model



(a) OSB



(b) MIPB

Figure 3.21: Bit and PSD allocation for the 6 line ADSL 2+ scenario in Figure 3.18 with 1% worst case FEXT model

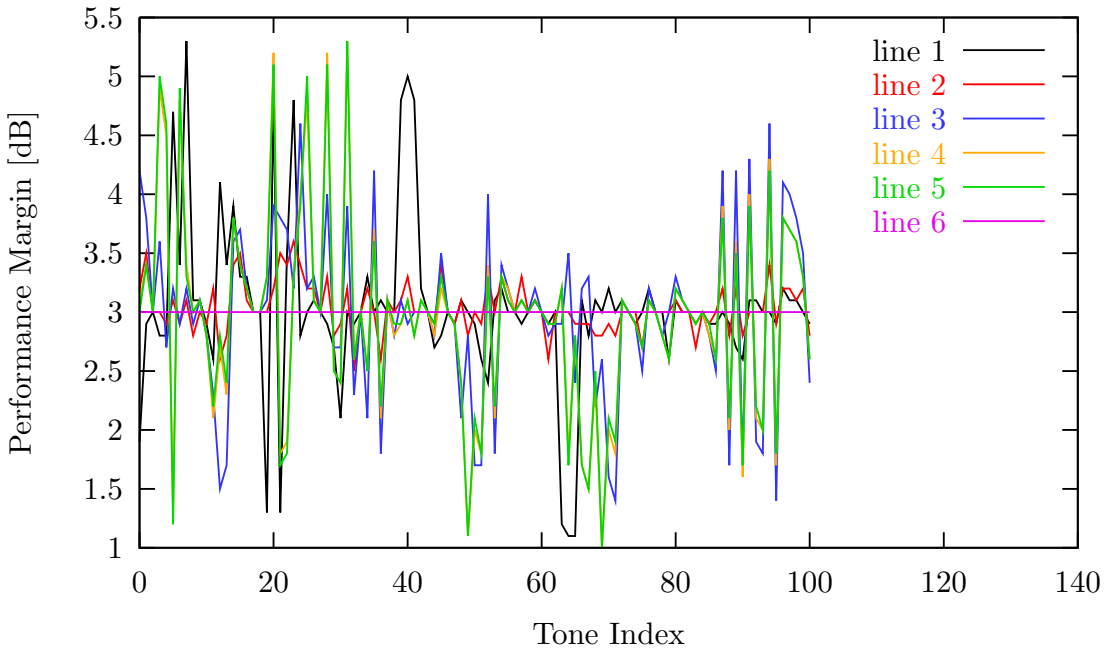


Figure 3.22: Performance margins on the first 100 tones for all lines from the network in Figure 3.18 using the Iterative Waterfilling Algorithm with 1% worst case FEXT model

OSB	$O(T^{\lambda}K\hat{b}^{N}T^{sys}N^{3})$
OSB with branch and bound	$O(T^{\lambda}K\hat{b}^{N}T^{sys}N^{3}/T^{CRF})$
ISB	$O(T^{\lambda}KN^{2}T^{sys}N^{3})$
MIPB	$O(B^{total}(T^{CF}KN^{2}+T^{sys}N^{3}))$

Table 3.13: Complexity of various bit-loading algorithms

3.5 Computational Complexity

In this section, the computational complexity of MIPB will be investigated and compared against other DSM algorithms. Introduction of parallelism into MIPB and existing DSM algorithms is also discussed, and the performance of the resulting parallelised algorithms is examined. In each of the following complexity calculations the $T^{sys}N^3$ term is common to each algorithm. This represents the solution of the per tone, $N \times N$ linear system in Equation 2.25 which is an $O(N^3)$ operation, along with it's associated time constant, T^{sys} The solution of this system is common to all level 2 DSM algorithms and the complexity analyses represent approximately how many times this system will need to be evaluted in order to find a solution.

3.5.1 OSB

Table 3.13 shows the complexity of a selection of bit-loading algorithms tested in this thesis. OSB is the most complex and hence the slowest algorithm examined. The per tone linear system, whose complexity is $T^{sys}N^3$ must be solved on each tone k. As the search for the maximum value of the Lagrangian requires an exhaustive search of all possible bit loading combinations on a particular tone, the complexity increases by a factor of $\hat{b}$ for each new user. In the OSB entry in Table 3.13, the constant T^{λ} represents the number of Lagrangian dual variable update steps either by bisection, sub-gradient or other methods.

3.5.2 OSB with Branch and Bound

For OSB with branch and bound, the constant T^{CRF} represents the complexity reduction factor introduced by the branch and bound operation. From the data presented in [37] in appears that $T^{CRF} \approx e^N$. Although this is a significant reduction in complexity, the $\hat{b}^N$ component grows much faster than T^{CRF} meaning that OSB with branch and bound is currently intractable for more than about 8 users.

3.5.3 ISB

ISB reduces the complexity of the search of the maximum Lagrangian such that it follows a square law in N [34]. T^{λ} once again represents the number of dual-update steps required for convergence of the power constraints.

3.5.4 MIPB

For each iteration of MIPB, the cost matrix must be recalculated for every line and tone in the bundle. The number of times this must be executed is KN. Each cost element update is $O(N)$ as it must sum the incremental power for each line. Thus the cost function step is approximately $O(KN^2)$.

Also on each iteration, the values of Δp on the tone on which the current bit was added must be updated. Each of these steps will be executed when a new bit is loaded which occurs will occur B^{total} times. The constant T^{CF} is the constant associated with the cost function update step.

3.5.4.1 MIPB Execution Profiling

The execution of MIPB was profiled for different problem sizes using the *gprof* [45] profiler. Figure 3.23 shows the total execution time required for the cost function update steps against the number of users in the system N. Referring to Table 3.13 this should be expected to follow approximately an aN^2 relationship. However, the effect of B^{total} has not yet been considered. B^{total} is the total number of bits loaded in the bundle and it should be approximately linear in

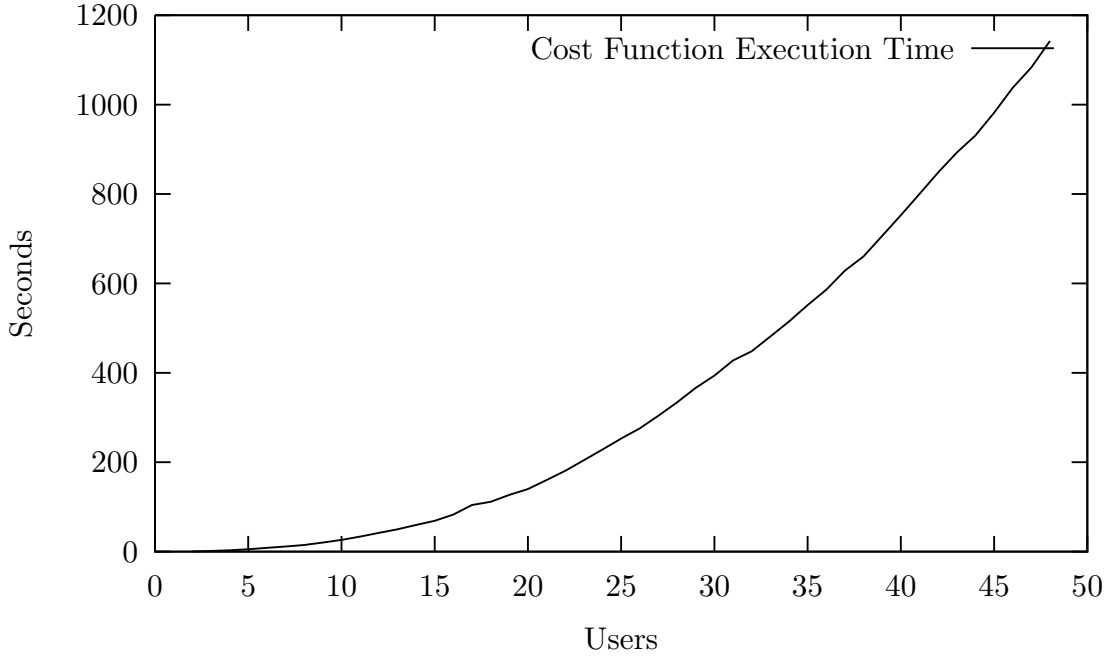


Figure 3.23: Cost function execution times against number of users for MIPB

N. B^{total} will determine the number of cost function update steps required and thus it is expected to introduce a smaller component of the execution time in N^3. Figure 3.24 shows the execution time alongside a curve of order $aN^2 + bN^3$. It is clear that this curve is an excellent fit for the data points and it confirms the complexity analysis given here. The values of the constants a and b are 0.302 and 0.004105 respectively, confirming that the cubic component is indeed small relative to the square component.

The execution times of the Δp update steps are illustrated in Figure 3.25. As with the cost function update times, it is expected that the effect of B^{total} in the complexity analysis should cause the $O(T^{\Delta p}N^3)$ factor to gain a small component of N^4. Figure 3.26 includes the curve fitted to the data points in the form $aN^3 + bN^4$. Again, this curve is an excellent fit for the data and seems to confirm the complexity analysis. The values of the constants a and b are 0.00139 and 2.16e-5 respectively.

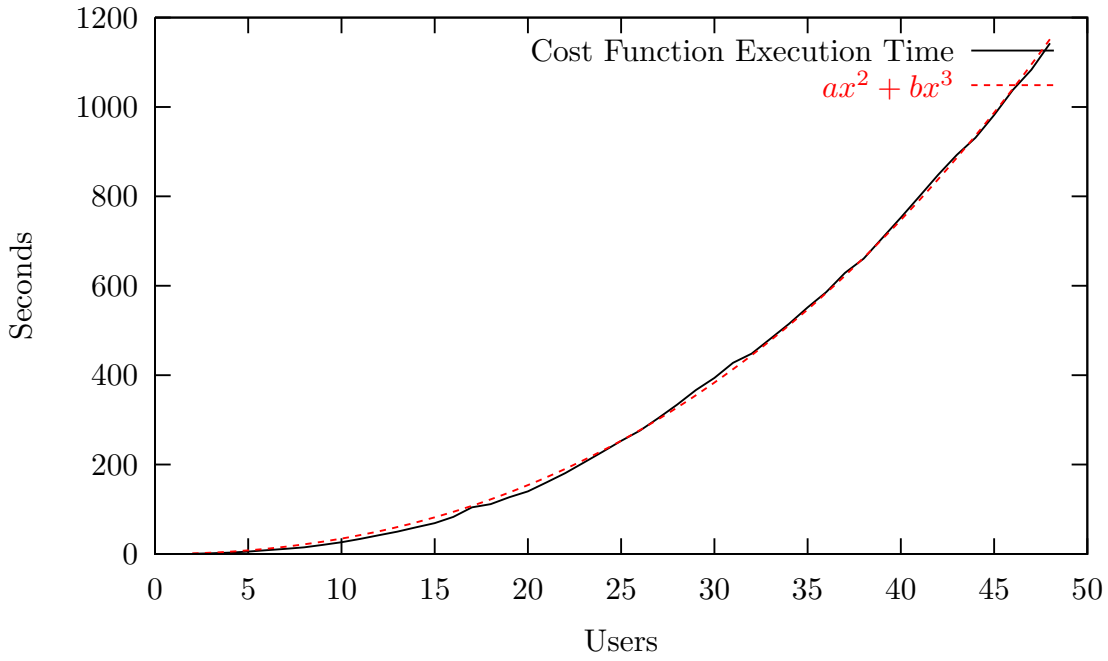


Figure 3.24: Cost function execution times against number of users for MIPB and the curve fitted of predicted execution order. $a = 0.302$ and $b = 0.004105$

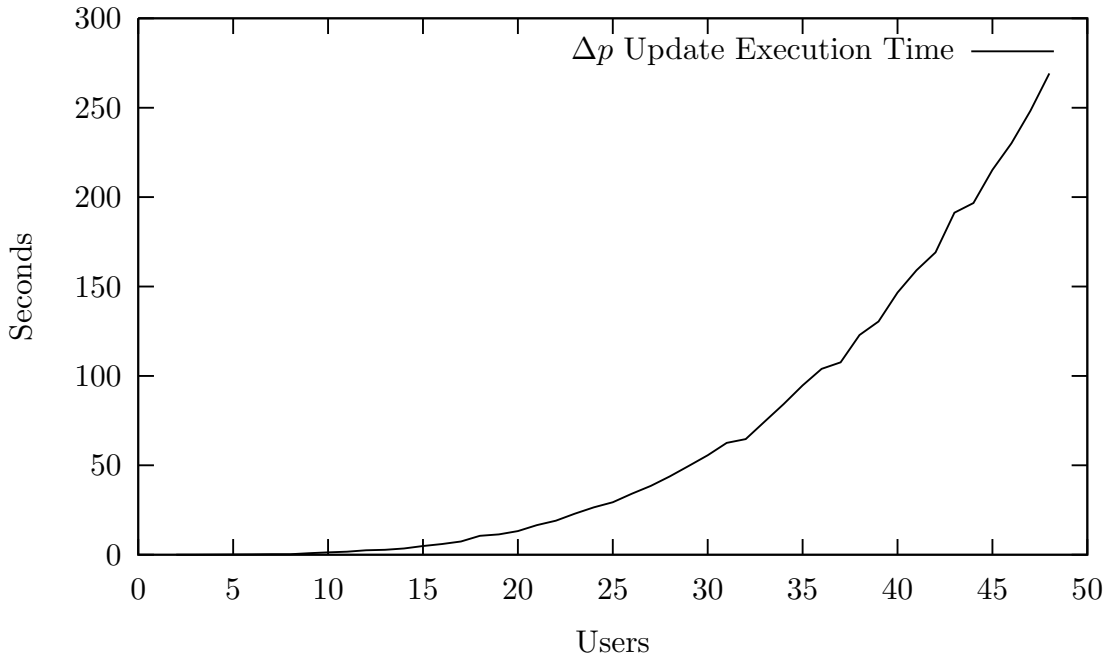


Figure 3.25: Δp update execution times against number of users for MIPB

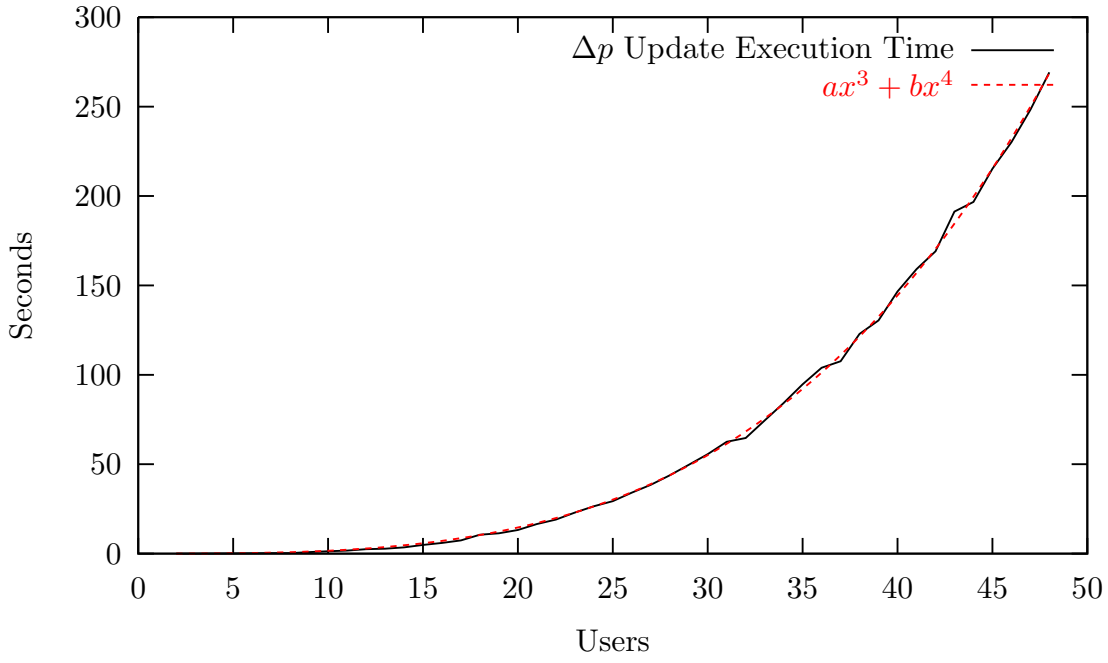


Figure 3.26: Δp update execution times against number of users for MIPB and curve fitted of predicted execution order. $a = 0.00139$ and $b = 2.16e - 5$

3.5.5 Relative Execution Times

In this section, the algorithm execution times for various DSM algorithms will be obtained by experiment for different numbers of users. The tests were performed on a 2GHz Intel Core 2 Duo machine, although at this stage each algorithm will run in a single thread of execution.

The algorithms OSB, OSB and branch and bound and ISB were tested alongside MIPB. For each of the Lagrangian dual type algorithms, the time to obtain a convergence of the power constraints with equal line weights were measured. The dual-update method utilised in the Lagrangian dual algorithms used is bisection. Bisection is not the most efficient update method, however it was found during experimentation that it is the only method that will reliably converge.

Figure 3.27 shows the execution times for these algorithms on a log scale of seconds against number of users. The extremely high complexity of OSB is apparent. Results were only obtained for 4 users for the purposes of this experiment, as 5 users would be prohibitively long. It is also clear that MIPB

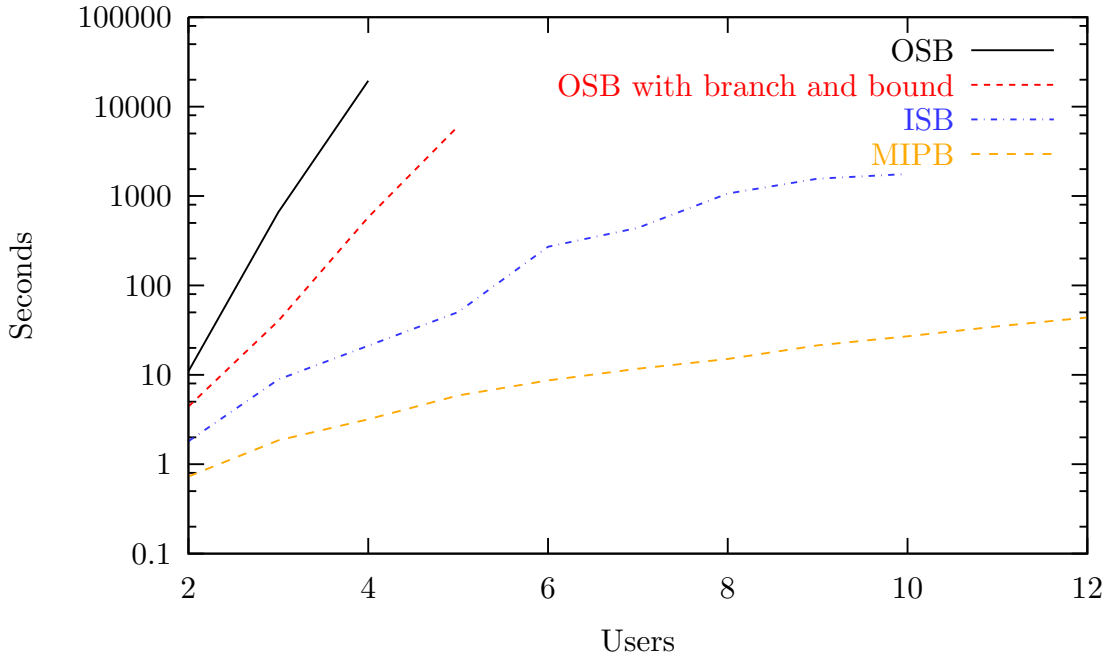


Figure 3.27: Execution times for various DSM algorithms against number of users

is the least computationally complex of these four algorithms by a significant margin. It also has the lowest gradient of each algorithm, demonstrating its scalability for larger problem sizes and tractability for large DSL scenarios.

3.6 Parallelisation of DSM Algorithms

For DSM to be achievable in real DSL systems with large numbers of lines, DSM algorithms must be runnable in a practically useful time. As most DSM algorithms are slow, they are prime targets for parallelisation, given the recent trend in computing hardware of including many CPU cores on one die and the even more recent topic of General Purpose Computing on Grapics Processing Units (GPGPU) programming.

In this section, the parallelisation of MIPB as well as as other popular centralised DSM algorithms is investigated. All of the following simulations were performed on a 16 x Itanium2 CPU system. The threads are implemented with posix threads for OSB with branch and bound and the *boost::threadpool* [46] library for ISB and MIPB.

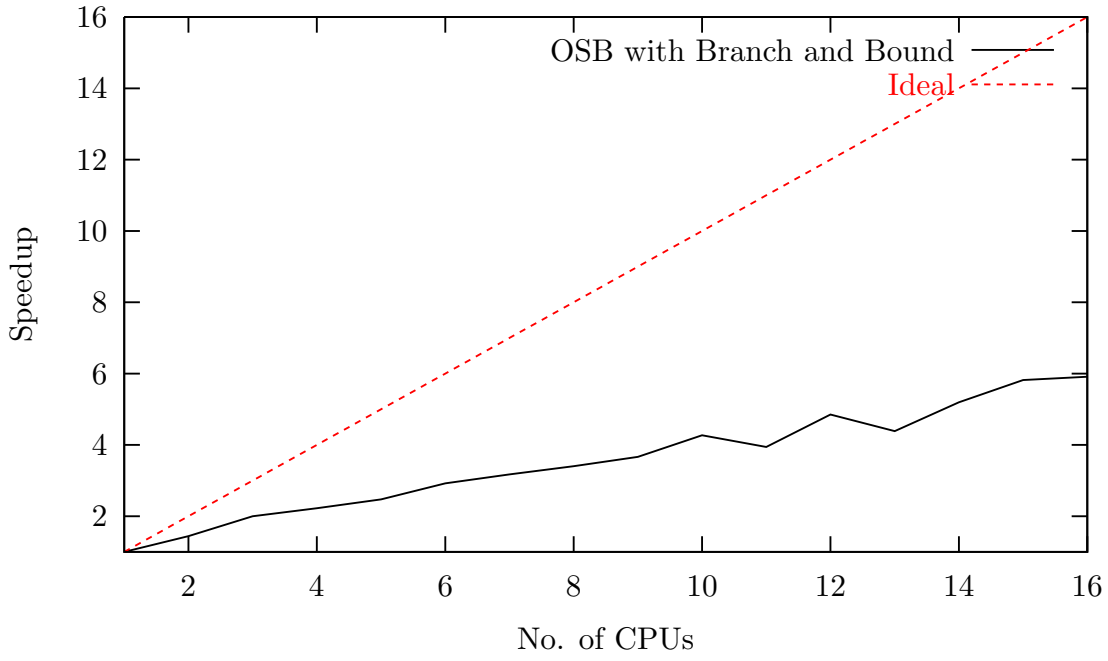


Figure 3.28: Speedup against number of CPUs for OSB with branch and bound

3.6.1 Parallelisation of Lagrangian Dual Algorithms

At the core of Lagrangian dual algorithms such as OSB and ISB, the maximum value of the Lagrangian must be be found or approximately found on each tone. These operations are independent of each other and thus they can be executed in parallel. This is illustrated in Algorithm 13 which shows a generic lagrangian dual decomposition, which could be solved by ISB,OSB or OSB with branch and bound. The inner loop which maximises the langrangian function can be independently executed for each tone.

Algorithm 13 Parallelised Lagrangian Dual Decomposition

repeat
  repeat
    for $k = 1 \dots K$ **do**
      Maximise Lagrangian on tone k ▷ Maximum of K parallelisable operations
    end for
    Update $\lambda_n, \forall n$
  until convergence
  Update $w_n, \forall n$
until convergence

Figure 3.28 shows that absolute speedup against the number of CPUs for a 4 line ADSL2+ downstream scenario for OSB with branch and bound. A moderate speedup is obtained, but the speedup obtained is less than that predicted by Amdahl's law, [1] due to a particular aspect of the branch and bound algorithm. In OSB with branch and bound, the solution from the previous tone is used as an initial guess for the maximum of the next tone. Given the fact the the transfer functions are approximately flat over a small frequency range, it is logical to conclude that the channel conditions, and thus the Lagrangian function, on tone k are very similar to those on tone $k+1$. By using the value of the optimal solution on tone k as an initial guess for the solution on tone $k+1$, a lower bound fairly close to the maximum value is obtained. This results in a lot of regions being discarded earlier in the branch and bound procedure, significantly speeding up convergence [37]. On tone 0, the branch and bound algorithm does not have an initial guess for the lower bound and thus is much slower than subsequent per tone optimisations. When multiple threads are used in OSB with branch and bound, fewer per tone optimisations can benefit from the speedup of the initial guess, thus reducing the potential speedup somewhat.

Figure 3.29 shows the absolute speedup obtained against the number of CPUs for a 6 line ADSL2+ downstream scenario for ISB. For up to 7 CPUs, an approximately Amdahl's Law-shaped relationship is obtained for $P = 0.99$, where P is the percentage of the algorithm that can be parallelised. Above 7 CPUs, a lower speedup than predicted by Amdahl's Law is obtained. It is suggested that this is due to increased lock contention inside the threadpool or possibly a non-ideal implementation, given the inherent difficulties of multi-threaded programming. The lock contention issues could be solved by utilising a more powerful thread abstraction, such as kernel threads, which can remove themselves from the run queue when not in use.

[1] $max\_speedup = \frac{1}{(1-P)+\frac{P}{M}}$, P = fraction parallelisable work, M = number of processors

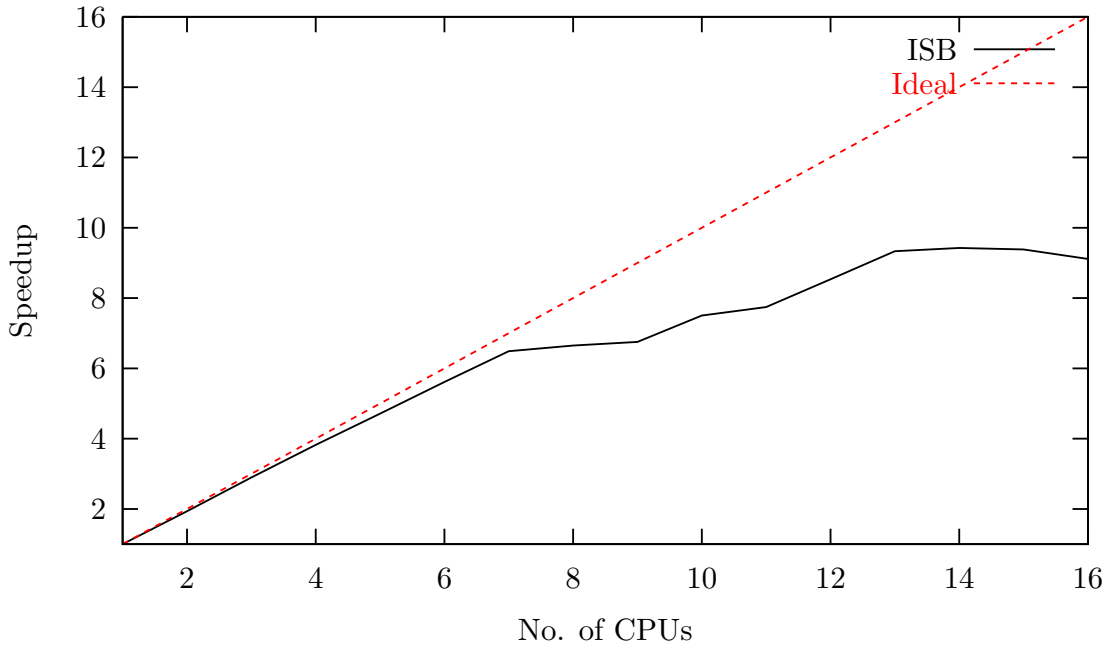


Figure 3.29: Speedup against number of CPUs for ISB

3.6.2 Parallelisation of MIPB

On each bit added during the operation of the MIPB algorithm, the new Δp values for adding a bit to each tone must be calculated to reflect the new state on the tone where the bit was added. This will result in N PSD vector calculations. Having determined the Δp values on that tone, the values of $wp(n)$ will be updated. As the cost matrix values depend on the values of $wp(n)$, when $wp(n)$ all the values in the cost matrix must be re-calculated, resulting in KN cost function calculations.

The PSD vector calculations and the cost function updates are independent of each other and thus can be executed parallel.

To investigate the parallelisation of MIPB, a 48 line ADSL2+ scenario was tested. In this scenario the line lengths and relative positions of the 6 line ADSL2+ network in Figure 3.18 are simply repeated 8 times. From the execution profile obtained for 48 lines, it was noted that approximately 98% of the total time was spent calculating cost function updates or per tone PSD calculations. It is assumed that this 98% can be parallelised to some degree and the remaining fraction cannot.

Percentage of Total	Cumulative Seconds	Total Calls	Function
44.4	635.69	1310791680	mipb::cf()
33.93	1121.56	1310791680	mipb::total_power_constraint_broken()
18.8	1390.76	2753808	psd_vector::calculate_psd_vector()
1.36	1410.17	1256505018	mipb::spectral_mask_constraint_broken()
98.49			

Table 3.14: Abbreviated execution profile from *gprof* for 48 DSL users calculated with MIPB

Given that approximately 2% of the algorithm must be executed serially, Amdahl's law can be used to predict the maximum speedup obtained from parallelisation of the remaining 98% of the algorithm. The parallelisation of MIPB was implemented using the *threadpool* model provided by the *boost::threadpool* library. The threadpool model assigns tasks to worker threads which are initialised when the programs starts, reducing the overhead of thread startup and teardown time. When a PSD recalculation is required on tone k, the new vector calculation for each line is added to the threadpool work queue. For the cost function calculations, each thread is delegated a subset of tones in the cost matrix to update.

Figure 3.30 shows the speedup obtained against number of CPUs utilised for a 48 line scenario using MIPB. Also shown on this graph is the ideal speedup possible with complete parallelisation and the maximum speedup given by Amdahl's law for 98% parallelisation.

In Figure 3.30 it can be seen that parallelised MIPB follows very closely the maximum possible speedup given by Amdahl's law up to approximately 8 CPUs. For greater numbers of CPUs, the speedup drops off somewhat. It is suggested that this drop off in speedup is due to limitations of the threadpool model when using userspace threads. The threadpool is initiated at the start of the algorithm

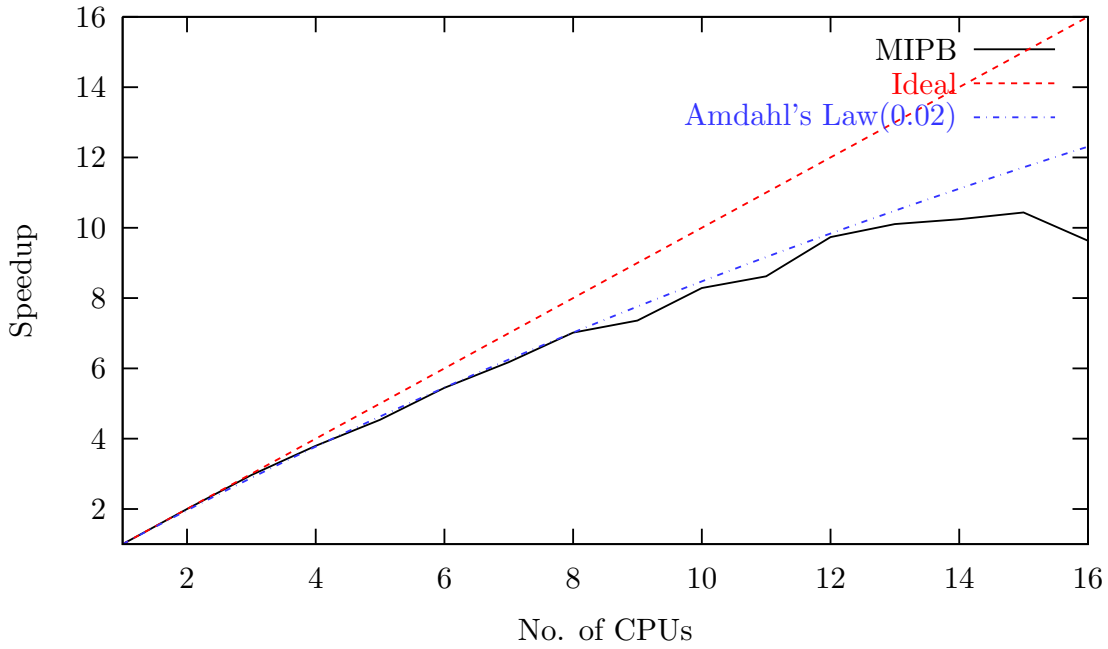


Figure 3.30: Absolute speedup for MIPB plotted against number of CPUs available. Also shown is the shape of Amdahl's Law for 2% unparallelisable work

and is not shutdown until all bits have been loaded. This is done to avoid thread startup and teardown overhead, which would otherwise be incurred. During this time each thread will constantly wake up and lock and then test the threadpool data structures to see if there is work to be done. As the number of threads increases, so does the lock contention on the threadpool structures, resulting in the lower than expected speedups. With userspace threads, there is no easy method to asynchronously schedule work but this problem may be solved by using more powerful kernel threads, with can be added asynchronously added to the run queue when required.

3.7 Conclusions

A low complexity centralised bit loading algorithm for dynamic spectrum management in xDSL systems was presented. Although the MIPB algorithm can at present only be used to find one data rate point (i.e. one rate per line), it is found that this rate point is near optimal and produces a solution very close to that of the OSB algorithm with equal weights for each user. For a range of widely different scenarios, MIPB outperforms existing bit loading algorithms such as ISB and IWF whilst exhibiting significantly lower complexity than existing centralised algorithms such as OSB and ISB.

A particularly attractive aspect of this algorithm is its very low complexity. This is shown to be several orders of magnitude faster that OSB and ISB, with much better scaling performance, meaning that results for large bundle sizes can be obtained that have previously only be achievable using the sub-optimal IWF.

Parallelisation of the MIPB algorithm was investigated and implemented, as well as that of existing centralised algorithms. These results show that large speedups can be obtained by partitioning the algorithms correctly and utilising modern multi-core computing hardware.

Chapter 4

Enhanced Multi-User Greedy Loading for DSM

In the previous chapter, a new algorithm called MIPB was presented which was shown to give a fair spectrum allocation that was close to the equal line weight optimum for OSB in a very short time and is scalable up to bundles of practical size. Unfortunately, a method to calculate arbitrary data rate points with MIPB was not developed.

This chapter details enhancements to the greedy algorithm [38] [39] [3] in order to calculate arbitrary data rate points in the rate regions. In particular, this chapter focuses on methods for improving the speed of operation of this algorithm and efficient parallelisation strategies. It also shows that the rate region performance of this algorithm is as close to optimal in realistic scenarios as to be of little practical difference.

4.1 Rate Regions

In this section, two methods for determining the rate regions using the greedy algorithm are presented and simulation results shown for various DSL networks.

In [3] the authors presented the multiuser greedy loading algorithm. The authors also briefly mentioned a method for calculating different data rates points using this algorithm. This method is shown in Algorithm 14.

Algorithm 14 Original Multiuser Greedy Rate finding algorithm [3]

repeat
  Execute Greedy Algorithm ▷ As in Algorithm 8
  for $n = 1 \ldots N$ **do**
    if $R_n < R_n^{target}$ **then**
      $w_n = w_n / \zeta$
    else $R_n > R_n^{target}$
      $w_n = w_n * \zeta$
    end if
  end for
until Rate Targets Met

Unfortunately, this method will lead to oscillations about the rate targets in most cases, unless the value of ζ is very small, which will in turn lead to very slow convergence behaviour. It is also not possible to know, a-priori, a value of ζ that will give good convergence behaviour.

Figure 4.1 shows the relationship between the value of ζ and the algorithm execution time for a 4 User ADSL2+ downstream scenario with the rate targets from Appendix A.1. The discontinuities in the graph are points where this algorithm does not converge.

4.1.1 Bisection of Rate Targets

By observing that the data rate on line n is a monotonic function of the value of w_n, it is proposed to utilise a bisection method to find the values of w_n on each line required to meet the data rate targets.

The bisection algorithm updates one value of w_n at a time and continues until all of the rate targets are within the specified tolerance. For each user, the greedy algorithm is executed to discover if the current value of w_n produces a rate above or below the rate target for that line. Having discovered this, the value w_n is either increased or decreased until a value on the opposite side of the rate target is found. If the rate target has not yet converged, the algorithm then proceeds to a traditional bisection.

In all of the examples used in this chapter, rate targets are set on all but one line, whose rate will therefore be maximised. The algorithm is illustrated in

Algorithm 15 Weight Bisection for Greedy Algorithm

$w_n = 1, \forall n$
repeat
  for $n = 1 \dots N$ **do**
    Execute Greedy Algorithm
    if $R_n < R_n^{target}$ **then** ▷ Found initial max
      $max = w_n$
      $w_n / = 2$
      repeat
        Execute Greedy Algorithm
        if $R_n < R_n^{target}$ **then**
          $max = w_n$ ▷ New max
          $w_n / = 2$
        else$R_n > R_n^{target}$
          $min = w_n$ ▷ Min found
        end if
      until min found
    else$R_n > R_n^{target}$ ▷ Found initial min
      $min = w_n$
      $w_n * = 2$
      repeat
        Execute Greedy Algorithm
        if $R_n < R_n^{target}$ **then**
          $max = w_n$ ▷ Max found
        else$R_n > R_n^{target}$
          $min = w_n$ ▷ New min
          $w_n * = 2$
        end if
      until max found
    end if
    $w_n = \frac{max+min}{2}$ ▷ Commence bisection between max and min
    repeat
      Execute Greedy Algorithm
      if $R_n < R_n^{target}$ **then**
        $max = w_n$
      else$R_n > R_n^{target}$
        $min = w_n$
      end if
      $w_n = \frac{max+min}{2}$
    until $|R_n - R_n^{target}| < R^{tol}$
  end for
until All Rate Targets Met

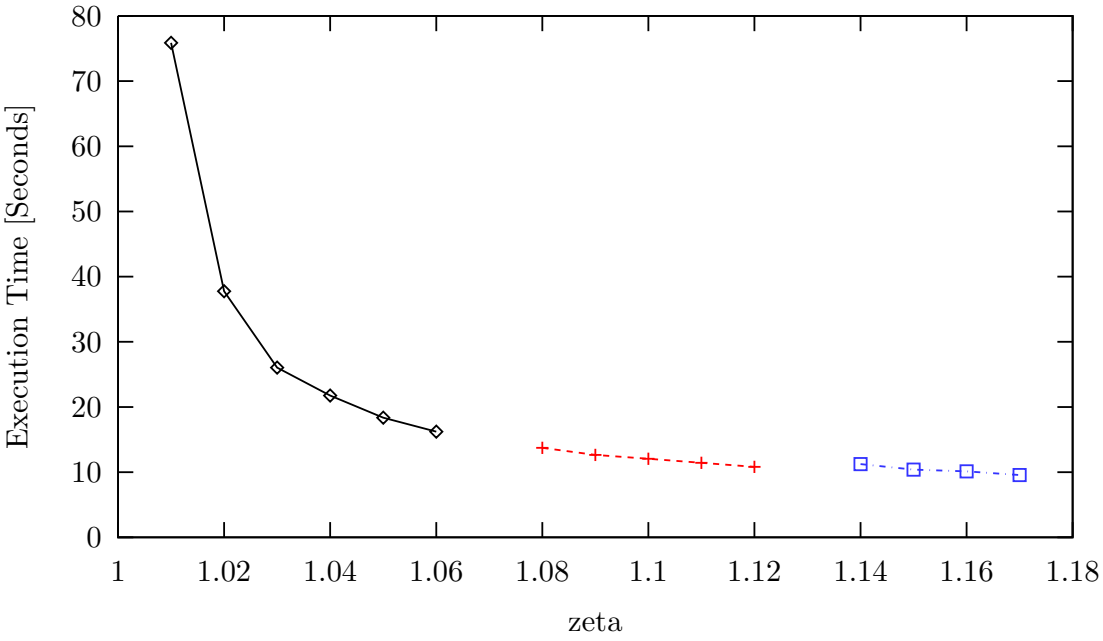


Figure 4.1: Relationship between execution times and the value of ζ used in Algorithm 14

Algorithm 15 and is used to determine the correct weight vectors to trace the rate regions calculated in the following section.

4.1.2 2 Users Near-Far ADSL Downstream

Once again, the scenario illustrated in Figure 2.6 be used to demonstrate the data rate performance of the Multi-User Greedy algorithm with rate bisection. The simulation parameters used are the same as those shown in Table 3.6.

The graph in Figure 4.2 shows the rate regions for this scenario achievable by OSB, ISB, IWF and finally, Greedy with weight bisection. The rate region performance of Greedy with bisection is seen to be quite close to the optimal solution given by OSB and slightly better than that of ISB. The performace of IWF is seen to be significantly poorer than the other three algorithms.

One particular data rate point is highlighted in table 4.1. In this example, the CO line rate is set at 600 bits per frame (2.4Mbps). At this rate point, ISB achieves a rate on the RT line within 5.7% of the optimal solution, IWF within

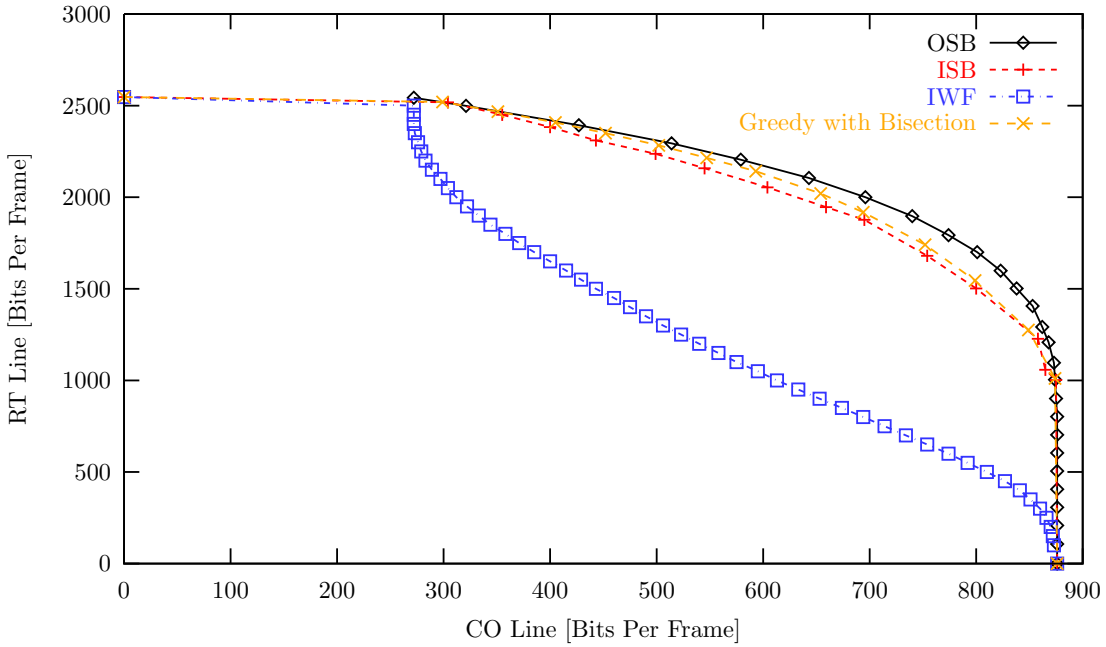


Figure 4.2: Comparison of rate regions for various loading algorithms compared to Greedy with bisection for the 2 User ADSL Downstream scenario in Figure 2.6

	OSB	ISB	IWF	Greedy
RT Line	2178	2054	1050	2142

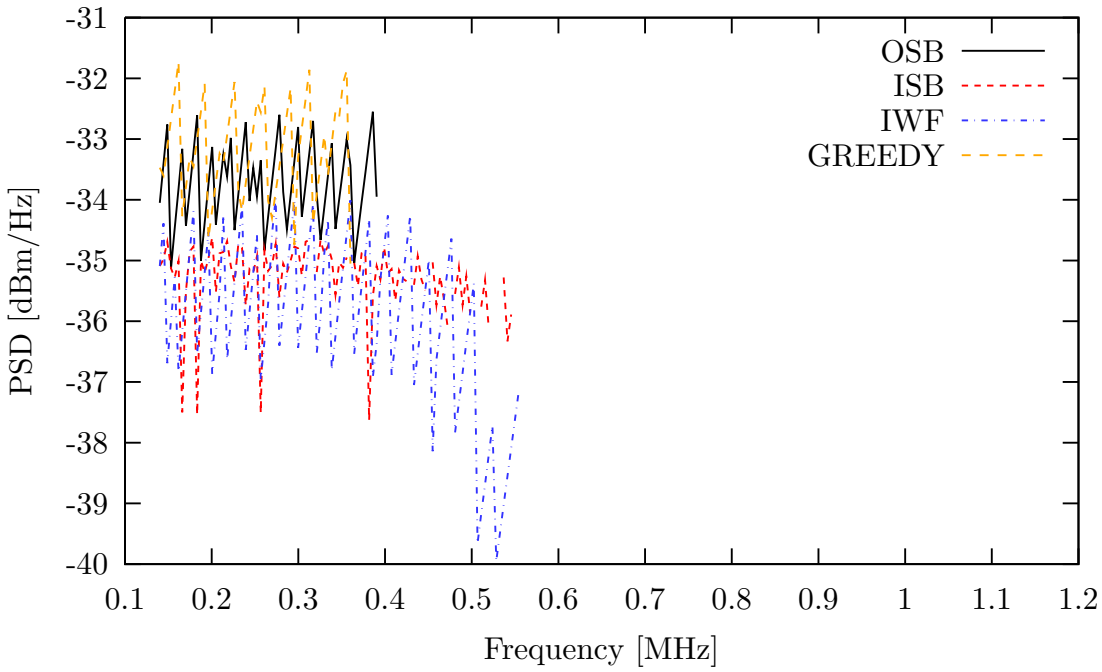
Table 4.1: Data rates for scenario in Figure 2.6 with the CO line rate fixed at 600 bits per frame

51.8% and greedy within just 1.7%.

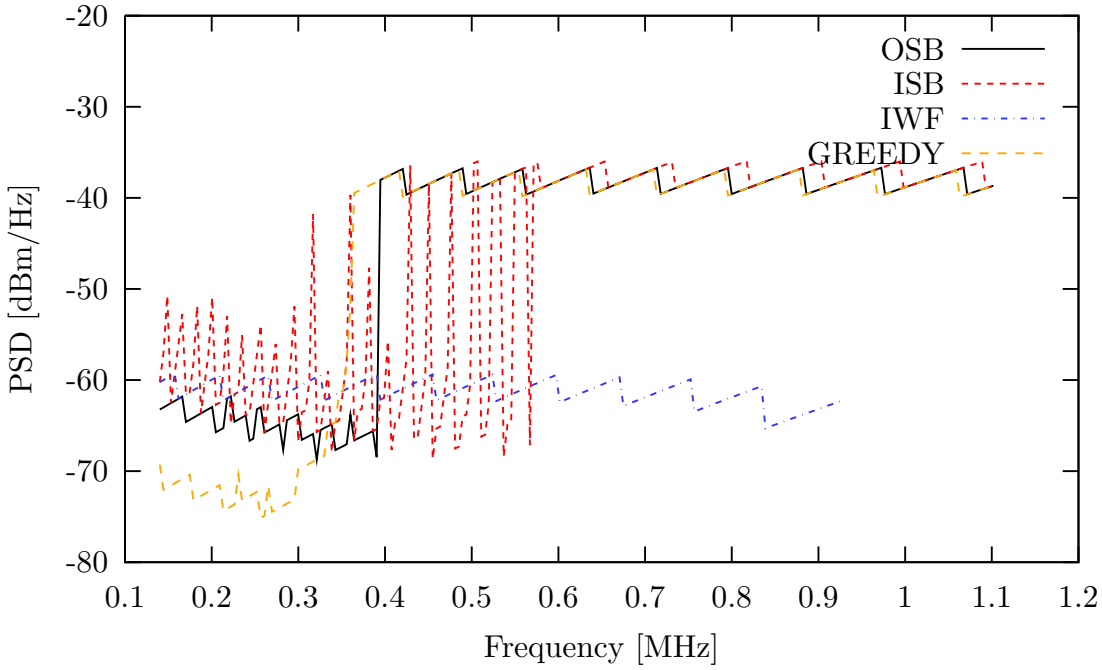
The spectra generated by each algorithm at this data rate point are shown on the same axis in Figure 4.3. On both the CO line and the RT line it can be seen that the greedy algorithm produces a spectrum which most resembles the optimal spectrum provided by OSB.

4.1.3 3 User VDSL 2 Upstream

The graph in Figure 4.4 shows the rate regions for the scenario in Figure 3.15 using the simulation parameters shown in Table 3.8. In these rate regions, the rates of user 1 and user 2 are traded against each other and the rate target for



(a) Line 1



(b) Line 2

Figure 4.3: Spectra generated for the scenario in Figure 2.6 by various algorithms with the CO Line data rate fixed at 600 bits per frame

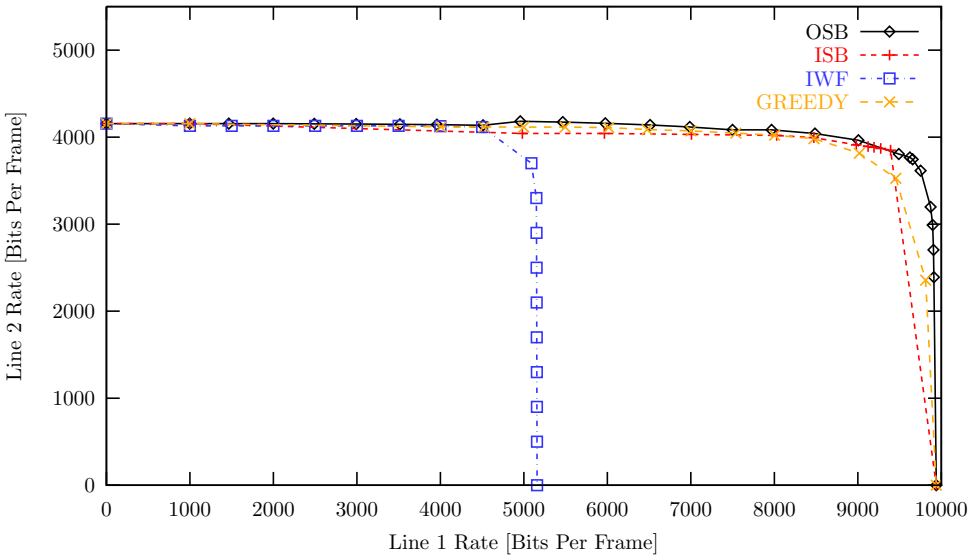


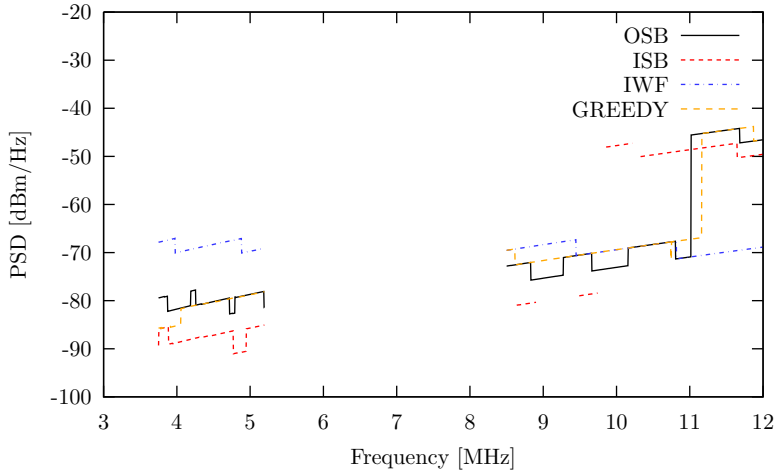
Figure 4.4: Comparison of rate regions for various loading algorithms compared to Greedy with bisection for the 3 User VDSL2 Upstream Scenario in Figure 3.15

the longest line is set close to its maximum date rate with no FEXT at 1400 bits per frame (5.6Mbps).

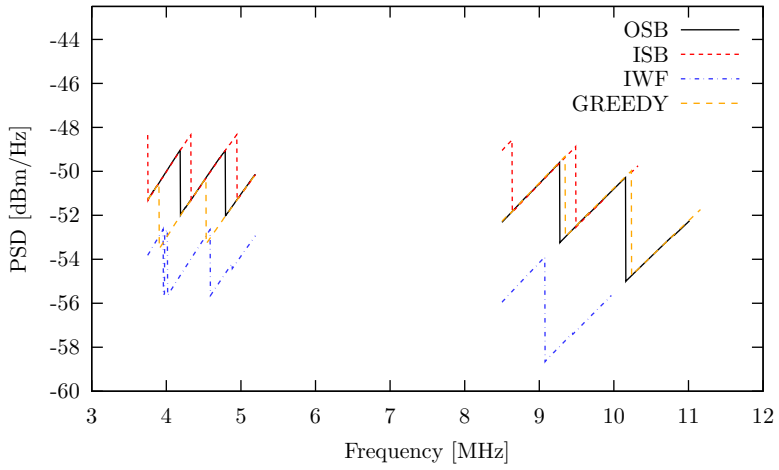
By inspecting the graph in Figure 4.4, the performance of the greedy algorithm is seen to be very close to that of OSB and ISB, whilst significantly outperforming IWF. The rate regions of OSB, ISB and the greedy algorithm are very similar in this case, tracing almost equally sized regions. IWF however, performs quite poorly. IWF can reach a maximum rate of 5150 bits per frame on line 1, almost 50% less than the maximum on line 1 for OSB.

The discontinuity in the ISB rate region at the end of its line 1 range is due to the fact that it is not possible to obtain a rate convergence for ISB in this region. It appears that there is a small region of non-convexity in w for ISB at this particular part of the rate region in this scenario.

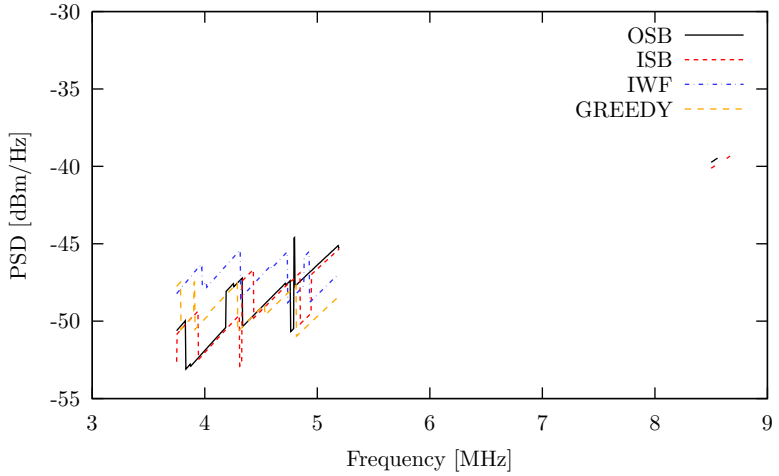
In Figure 4.5, the spectra for a particular data rate point on the rate region are shown. The data rates on lines 1 and 3 are fixed at 5000 and 1400 bits per frame respectively. The resulting data rates on line 2 are shown in Table 4.2. Once again, it is found that the greedy algorithm performs close to the optimal solution and is slightly better than ISB and significantly better than IWF. The spectrum on each line are illustrated in 4.5. It is clear that the spectrum generated by the



(a) Line 1



(b) Line 2



(c) Line 3

Figure 4.5: Spectra generated for the scenario in Figure 3.15 by various algorithms. The data rate on line 1 and line 3 are fixed at 5000 and 1400 bits per frame respectively

	OSB	ISB	IWF	Greedy
Line 2	4181	4042	3080	4124

Table 4.2: Data rates for scenario in Figure 3.15 with data rates of line 1 and 3 fixed at 5000 and 1400 bits per frame

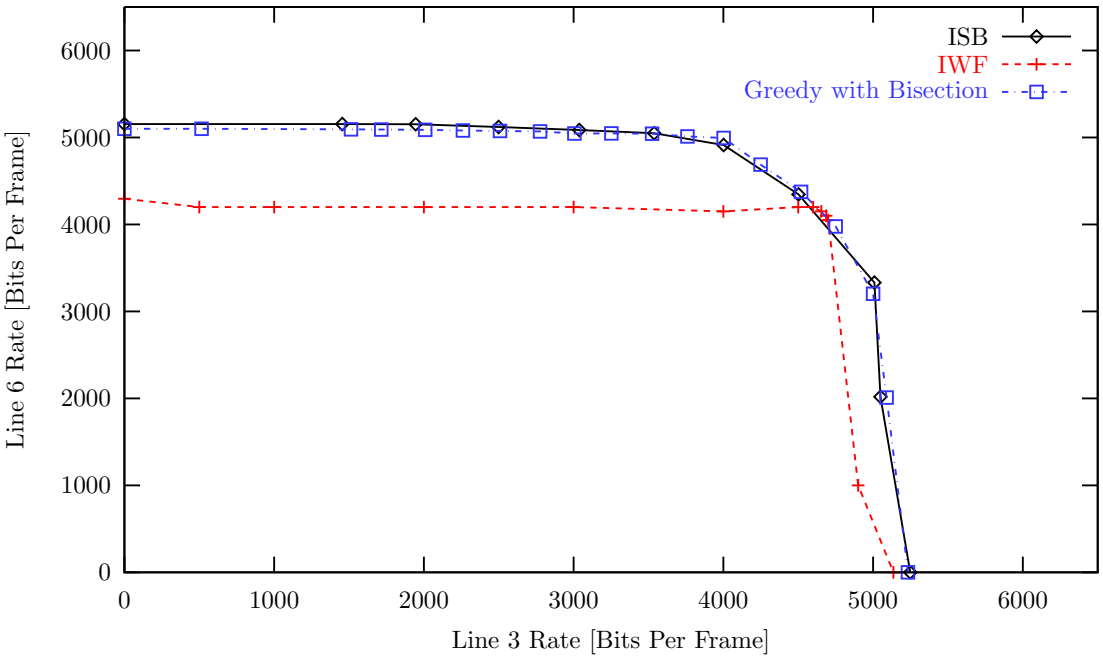


Figure 4.6: Comparison of rate regions for various loading algorithms compared to Greedy with bisection for the 3 User VDSL2 Upstream Scenario in Figure 3.18

greedy algorithm is very close to the optimal solution in this case and much closer than that of ISB or IWF.

4.1.4 6 User ADSL 2+ Downstream

The graph in Figure 4.6 shows the rate regions for the scenario in Figure 3.18 using the same simulation parameters as shown in Table 3.10. In the rate regions shown, the rate combinations of user 3 and user 6 are shown while the rate targets of the other lines are fixed at 2600 bits per frame (10.4Mbps), 2000 bits per frame (8 Mbps), 5300 bits per frame (21.2 Mbps) and 7100 bits per frame (28.4 Mbps) respectively.

In Figure 4.6, the greedy algorithm is shown alongside IWF and ISB. Due to the high complexity of OSB, it would have been prohibitively time consuming to obtain a rate region for OSB for this scenario so it is not included.

The greedy algorithm is shown to have very similar performace to ISB and significantly better performance that IWF. By deduction from the previous rate regions, it is reasonable to assume that once again the greedy rate region is comparable to that of the optimal solution.

Figure 4.7 shows the spectra produced by ISB, IWF and the greedy algorithm for this scenario. It is observed that the PSDs of ISB and greedy are very similar. IWF produces similar spectra on lines 1,2 and 4 but shows a marked difference on lines 3, 5 and 6. These differences in the PSDs account for the poor performance of IWF at this data rate point.

4.2 Sub-Gradient Weight Search

An alternative rate searching algorithm to the bisection method is a sub gradient search. Stated simply, the sub-gradient search adjusts the current value of the weight by some fraction of the current distance from the rate target.

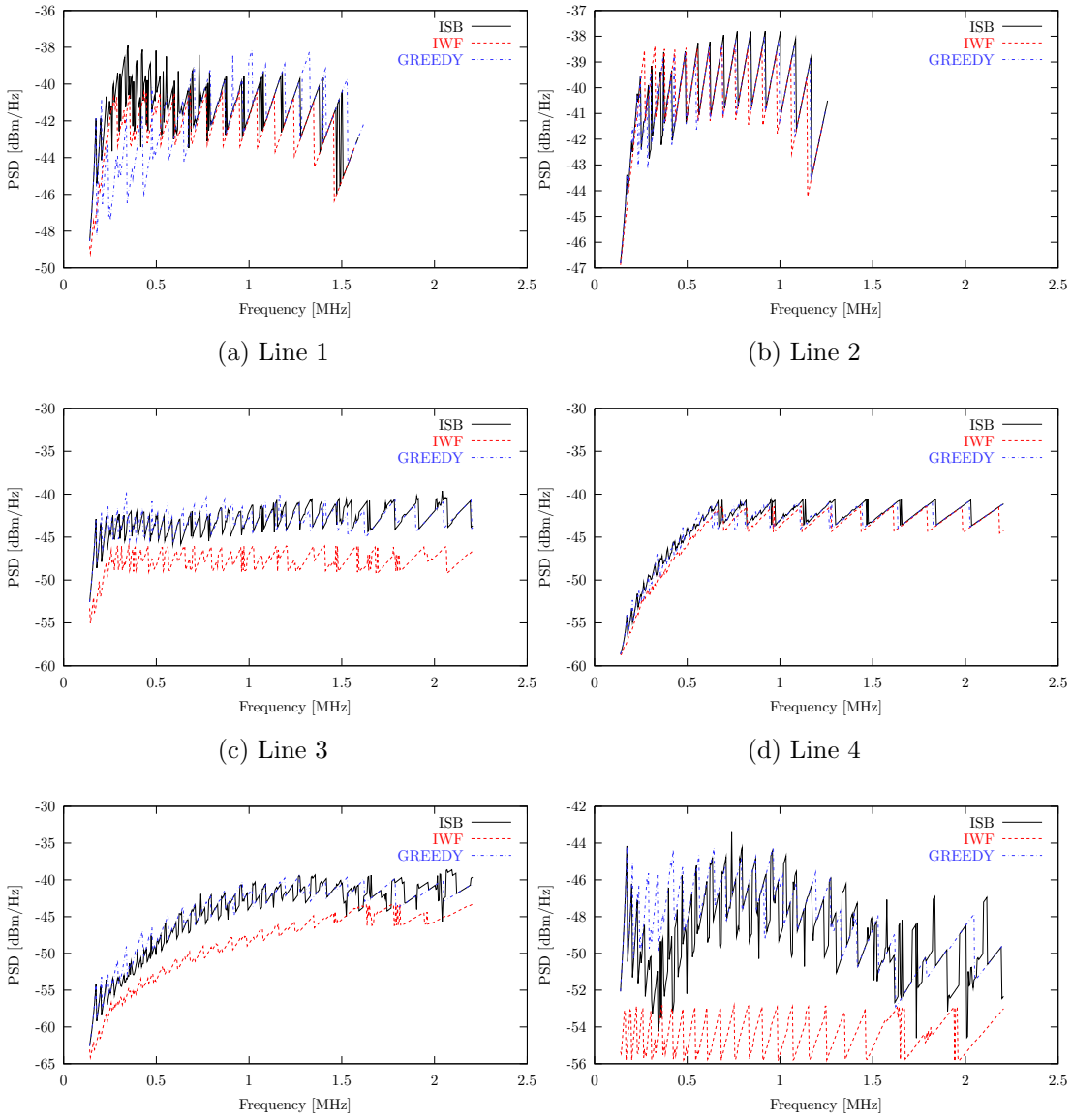
$$w_n^{+1} = w_n + \epsilon(R_n - R_n^{target}) \tag{4.1}$$

The main advantage of a sub-gradient method over bisection is that the weights of all users can be updated simultaneously, potentially requiring a lower number of iterations to converge.

Algorithm 16 Sub-Gradient Weight Search for Greedy Algorithm

$w_n = 1, \forall n$
repeat
    Execute Greedy Algorithm
    $w_n^{+1} = w_n + \epsilon(R_n - R_n^{target}), \forall n$
until All Rate Targets Met

The disadvantage is that the correct step is generally not known a-priori. A very high step size ϵ can result in fast convergence but may cause oscillations



(a) Line 1

(b) Line 2

(c) Line 3

(d) Line 4

(e) Line 5

(f) Line 6

Figure 4.7: PSDs generated by ISB, IWF and Greedy for the 6-User ADSL2+ scenario in Figure 3.18

	4 User		8 User
Line 2	1900 (7.6 Mbps)	Line 2	800 (3.2 Mbps)
Line 3	4700 (18.8 Mbps)	Line 3	2500 (10 Mbps)
Line 4	4300 (17.2 Mbps)	Line 4	3000 (12 Mbps)
		Line 5	4000 (16 Mbps)
		Line 6	3800 (15.2 Mbps)
		Line 7	900 (3.6 Mbps)
		Line 8	1800 (7.2 Mbps)

Table 4.3: Rate targets for 4 and 8 User ADSL2+ scenarios used for results in Figure 4.8

around the point of convergence, effectively rendering the algorithm useless. A very low step can guarantee convergence, but may result in a very slow convergence time. The sub-gradient search algorithm used along with the greedy bit-loading algorithm is shown in Algorithm 16.

In Figure 4.8, a comparison between bisection and sub gradient search for the greedy algorithm is illustrated for two DSL scenarios. The first is a four user ADSL2+ downstream scenario which uses the first four lines of the 6-User ADSL2+ scenario shown in Figure 3.18. The second is an eight user ADSL2+ downstream scenario which uses the same configuration as Figure 3.18 with the first two lines repeated. The rate targets for each line in the two scenarios are shown in Table 4.3. In both cases, the rate of Line 1 is left to be maximised. Illustrated in Figure 4.8 is the number of executions of the greedy algorithm required to complete the rate search for both scenarios against the step size ϵ. Two straight lines are also shown in Figure 4.8 representing the number of executions required by the bisection algorithm for each scenario from Algorithm 15. The decrease in convergence time as the step size increases is evident for both scenarios. However, for both scenarios the final step size values shown are the limit of stability for the sub gradient search above which the rate search does not converge.

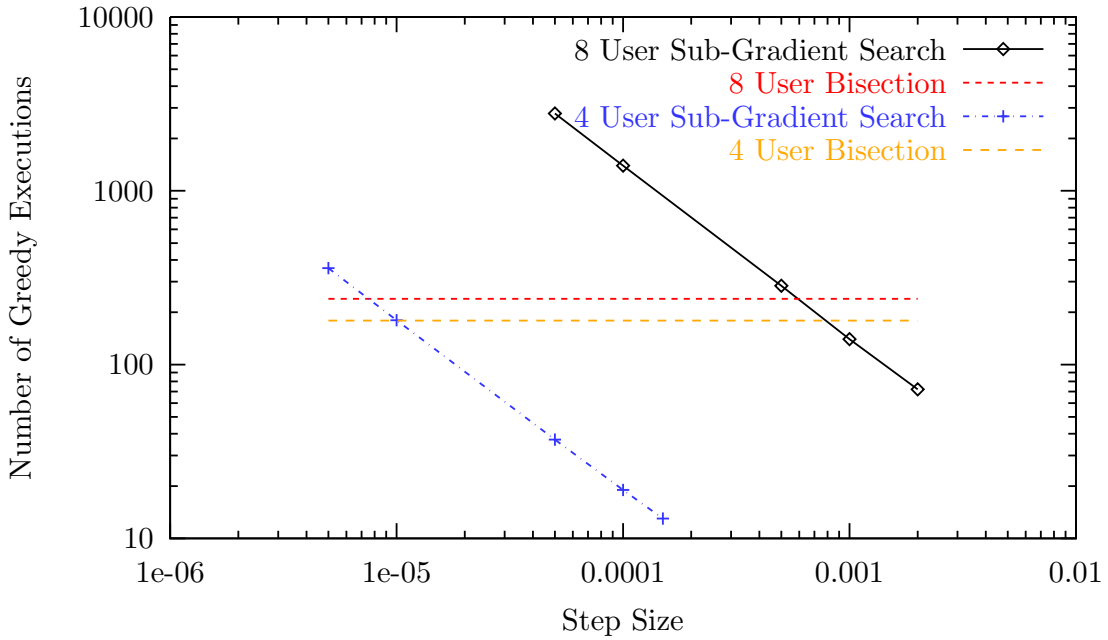


Figure 4.8: Number of executions of the greedy algorithm required for two scenarios against the step size ϵ. Also shown is the number required using the bisection method for each scenario

It also clear from Figure 4.8 that there is no universally good value for the step size, so it may be a risk to choose sub gradient search over bisection. problem will be presented.

4.2.1 Adaptive Sub-Gradient Search

In order to combat the problem of step size selection in a sub-gradient rate search, in this section, a new algorithm is presented. This algorithm utilises a sub-gradient search with an adaptive step size. Simulation results are presented and a significant performance advantage over bisection is observed without stability problems.

Figure 4.9 illustrates a sub-gradient rate search for two different 3-User scenarios. In this case, there are rate targets set for line 1 and 2 and the rate on the third line is subsequently maximised. Shown in Figure 4.9 are both a well behaved sub-gradient search and a badly behaved one, which exhibits oscillatory behaviour. The boxes represent the range in which the rate targets are within

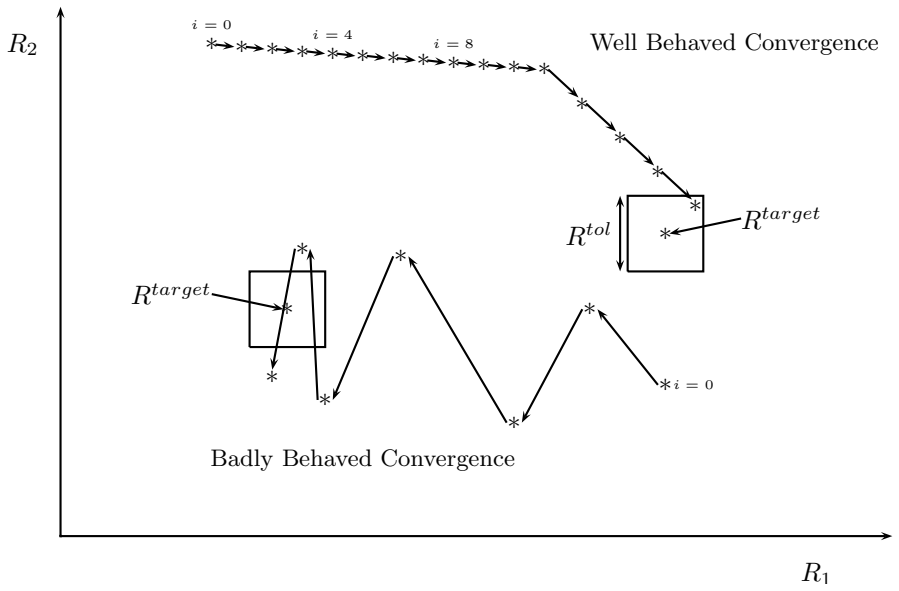


Figure 4.9: Illustration of a sub-gradient search of the weight vector w

an acceptable tolerance. With the standard sub-gradient search algorithm from Algorithm 16 it is not possible to know a-priori whether the value of ϵ will lead to slow well behaved convergence, fast convergence or not converge due to oscillations.

A real example of sub-gradient rate search iterations is illustrated in Figure 4.10. Since this can only be visualised in two dimensions (see Section 2.9.1), the scenario chosen is the 3-User Upstream VDSL scenario from Figure 3.15. The rate targets are set for lines 1 and 3 as 8000 and 1400 bits per frame respectively. The rate of the line 2 will be maximised. The rectangle in Figure 4.10 represents a tolerance of 80 bits per frame on each line.

Two values of ϵ are utilised in this scenario to demonstrate well behaved convergence and oscillation. Both traces in Figure 4.10 start at the bottom rightmost point on the graph before moving towards the rate target. The trace which utilises a value of $\epsilon = 1e - 5$ is seen to exhibit slow but direct convergence to the rate targets. The trace which utilises a value of $\epsilon = 0.0085$ is seen to oscillate around the rate targets and in fact does not converge.

An algorithm which attempts to overcome the step size selection problem for

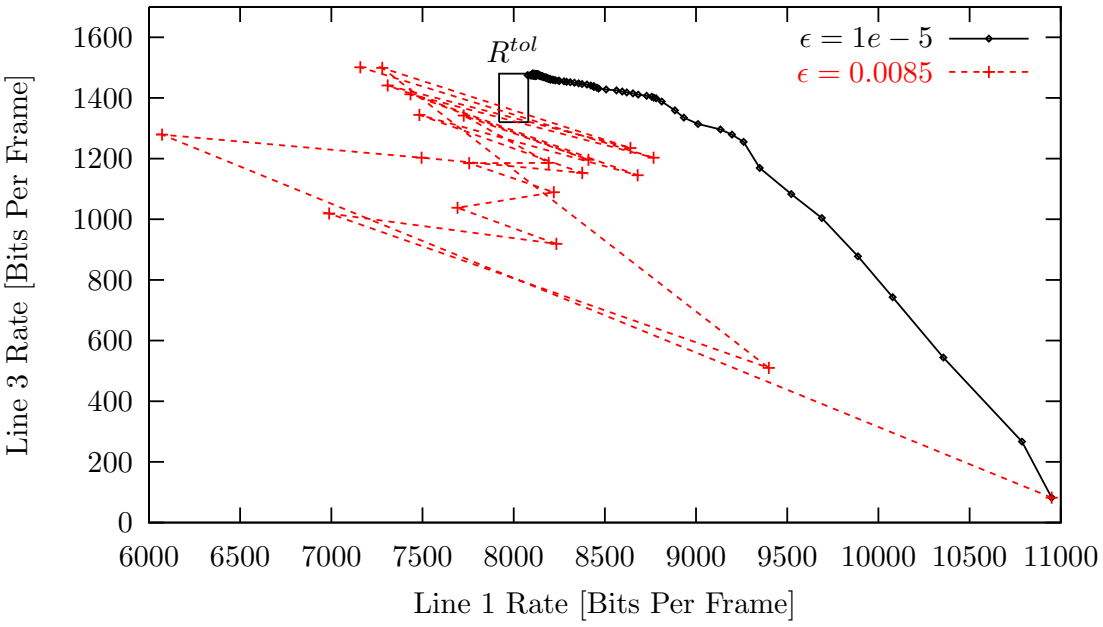


Figure 4.10: Illustration of iterations of sub gradient rate search for 3-User Upstream VDSL scenario in Figure 3.15

sub-gradient search is shown in Algorithm 17. The algorithm works by starting from a very low value of ϵ and increasing it until oscillatory behaviour is detected. Oscillatory behaviour is defined as two lines moving from one side of their rate target to the other during one iteration of the algorithm. Once oscillatory behaviour is detected, the algorithm moves back one iteration to the previous rate point and starts again with a smaller step size.

Figure 4.11 illustrates the operation of Algorithm 17 for the 3-user upstream VDSL scenario in Figure 3.15. The rate targets and tolerances are identical as those used in the sub-gradient searches in Figure 4.10. It is clear from this trace that the adaptive sub-gradient algorithm reaches the rate targets in a much more efficient manner than that of the small ϵ sub-gradient search in Figure 4.10, without exhibiting the oscillatory behaviour given by a large value of ϵ.

Figure 4.12 shows the number of greedy executions required by the bisection algorithm and by the adaptive sub-gradient method against the number of users for a downstream ADSL2+ scenario. For this experiment, the line parameters used are repeating patterns of the 6-User scenario in Figure 3.18. The rate targets

Algorithm 17 Adaptive Sub-Gradient Weight Search for Greedy Algorithm

$w_n = 1, \forall n$
$\epsilon = 1e-7$
Execute Greedy Algorithm
$w_n^{+1} = w_n + \epsilon(R_n - R_n^{target}), \forall n$
repeat
  $osc = 0$
  Execute Greedy Algorithm
  for $n = 1 \dots N$ **do**
    if $R_n > R_n^{target} \cap R_n^{-1} < R_n^{target}$ **then**
      $osc = osc + 1$
    else if $R_n < R_n^{target} \cap R_n^{-1} > R_n^{target}$ **then**
      $osc = osc + 1$
    end if
  end for
  if $osc \geq 2$ **then** ▷ Two or more lines rates are oscillating
    $\epsilon = \epsilon \div 2$ ▷ Reduce step size by half
    $w_n^{+1} = w_n^{-1} + \epsilon(R_n^{-1} - R_n^{target}), \forall n$
    ▷ Use previous value of w and R with new ϵ
  else
    $\epsilon = \epsilon \times 2$ ▷ Increase step size by two
    $w_n^{+1} = w_n + \epsilon(R_n - R_n^{target}), \forall n$
  end if
until All Rate Targets Met

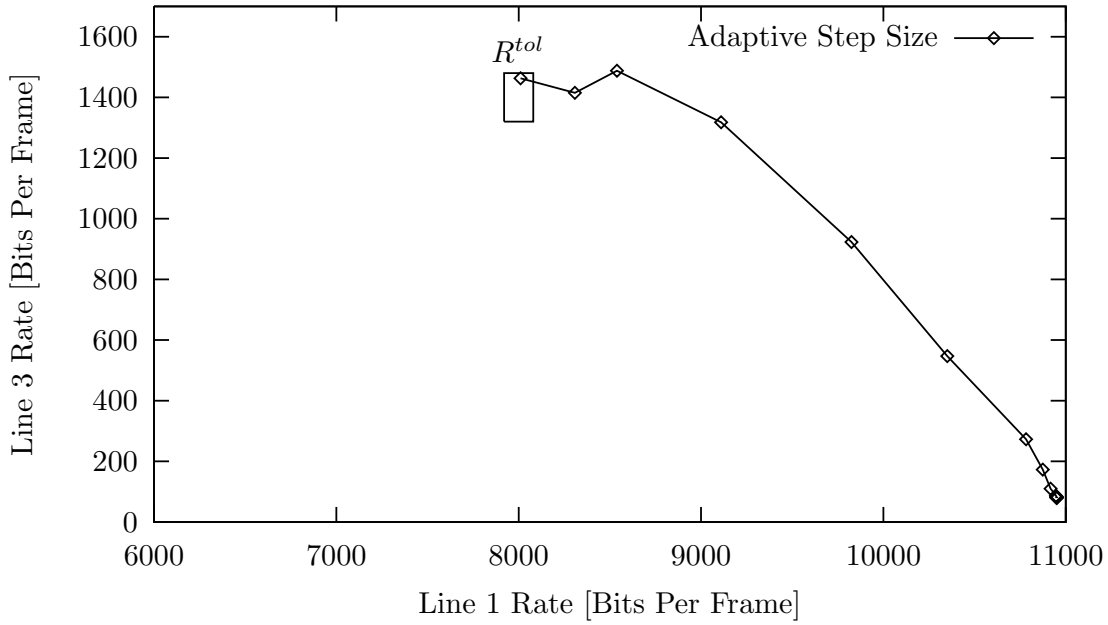


Figure 4.11: Illustration of iterations of the adaptive sub-gradient rate search for 3-User Upstream VDSL scenario in Figure 3.15

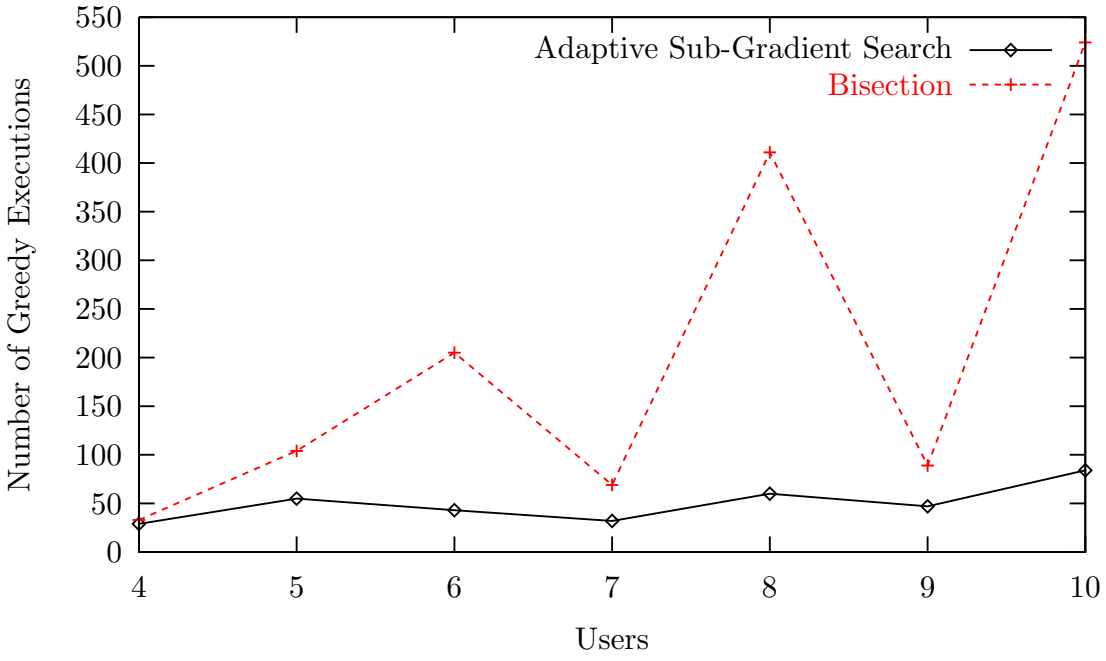


Figure 4.12: Number of greedy executions required for a bisection rate search and the adaptive sub-gradient rate search

used are shown in Appendix A.1. It is clear that the adaptive sub-gradient algorithm significantly outperforms the bisection method for all the scenarios tested. It also appears to scale more gracefully than bisection with the number of users, meaning that the speedups will be larger for increasing number of users. The peaks and troughs in figure 4.12 are caused by the particular rate targets chosen for each scenario which take varying numbers of iterations to converge. Different sets of rate targets would result in the peaks and troughs being distributed differently.

4.3 PSD Vector Caching

Disregarding the initial cost matrix calculation, the greedy algorithm requires approximately

$$B(O(N^4) + O(K))) \tag{4.2}$$

operations, where B is the total number of bits loaded. The $O(N^4)$ term is the cost of calculating the new PSD values on a particular sub-channel. This is an $O(N^3)$ operation, executed N times.

The complexity expression in expression 4.2 above is only true for one iteration of the greedy algorithm. When a rate target search is carried out by either bisection or sub-gradient search, the greedy algorithm will be executed many times in succession. The complexity in this case will be approximately $T^w B\dot{(}O(N^4) + O(K)))$ where T^w is the number of iterations required in the rate search.

Considering the fact that the values of the channel gains and crosstalk gains are identical between iterations of the greedy algorithm (i.e. the system does not change between executions), it is reasonable to assume that some of the same PSD vectors will calculated for a particular tone in separate iterations of the rate search. In other words, the algorithm will need to calculate the PSD vector for a particular bit vector that has been calculated in a previous iteration of the algorithm.

Given that the PSD vector calculation is a $O(N^3)$ operation, it is posited that there might be a significant speed improvement possible if the algorithm were to cache the values of the PSD vectors that is has already calculated.

Figure 4.13 shows an illustration of the algorithm operation visualised as a flow diagram. The update w values step in the diagram can be a per user bisection or a sub-gradient search. As shown in the diagram, the update Δp step checks if the current PSD to be calculated has already been calculated and therefore present in the cache. If the result has not been cached it is added to the cache.

Figure 4.15 shows the execution times obtained using the greedy algorithm with a bisection rate search against number of users with and without PSD

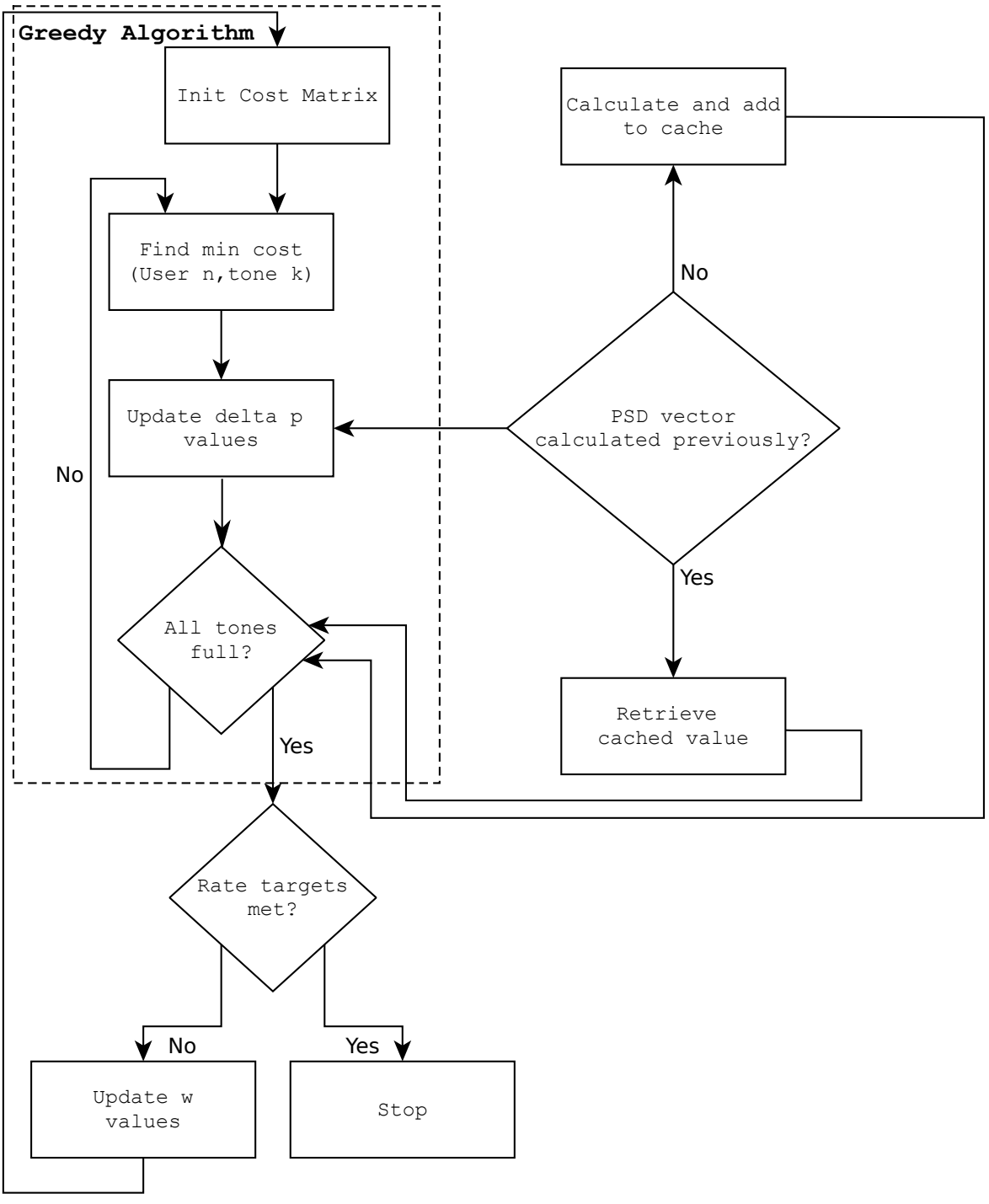


Figure 4.13: Flow chart showing operation of the greedy algorithm with rate search and PSD caching

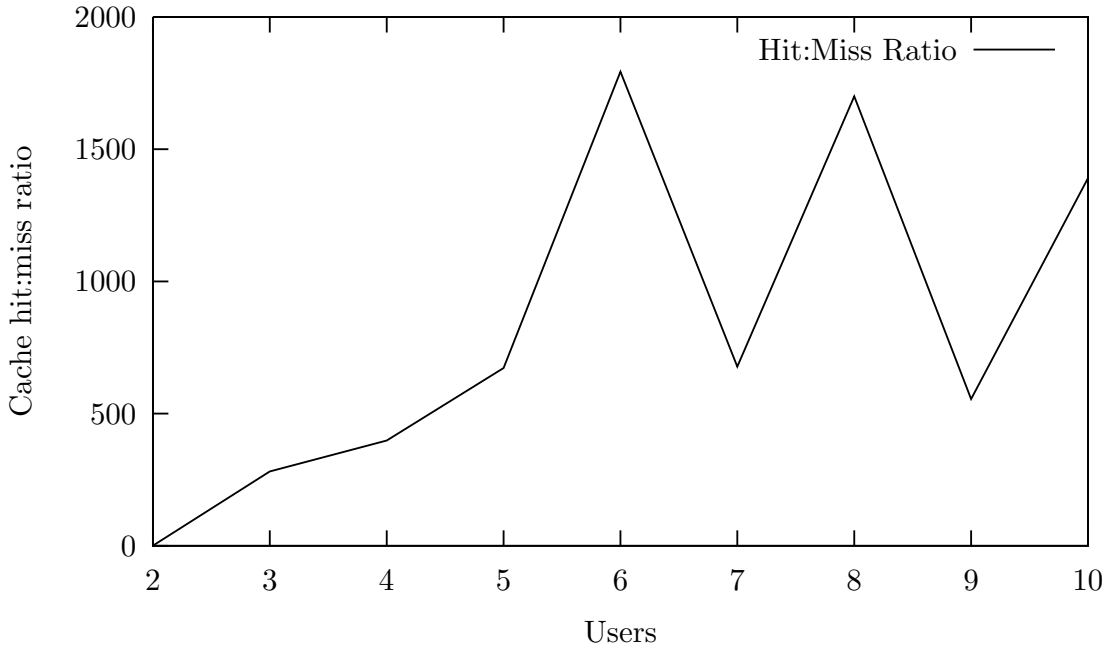


Figure 4.14: Cache hit:miss+collision ratio for the PSD Vector cache against number of ADSL users obtained from profiling the cache during simulation

caching. The rate targets for each set of users in shown in Appendix A.1. For ease of implementation, a fixed size cache with a maximum total of 38.4 million entries was used, that is 80000 entries per tone on 480 tones (ADSL 2+ downstream). The cache utilises a hash table on each tone with a fixed number of buckets (hash table entries) set at 80000. This reduces the complexity of the implementation but will inevitably lead to some collisions in the hash table. The cache uses a Most Recently Used (MRU) replacement strategy, where the most recent value is entered into the cache if there is a collision. This is chosen because it is expected that small changes in the weights during the rate search will lead to similar behaviour of the greedy algorithm, resulting in a fewer total number of collisions in the cache.

Figure 4.15 shows that a significant speedup is obtained when using PSD Vector caching. Figure 4.16 shows the absolute speedup obtained against the number of users, which is as large as 14 times faster for 8 users. The number of greedy executions required in each case is also illustrated. It is expected that a larger speedup would be observed when a larger number of greedy executions

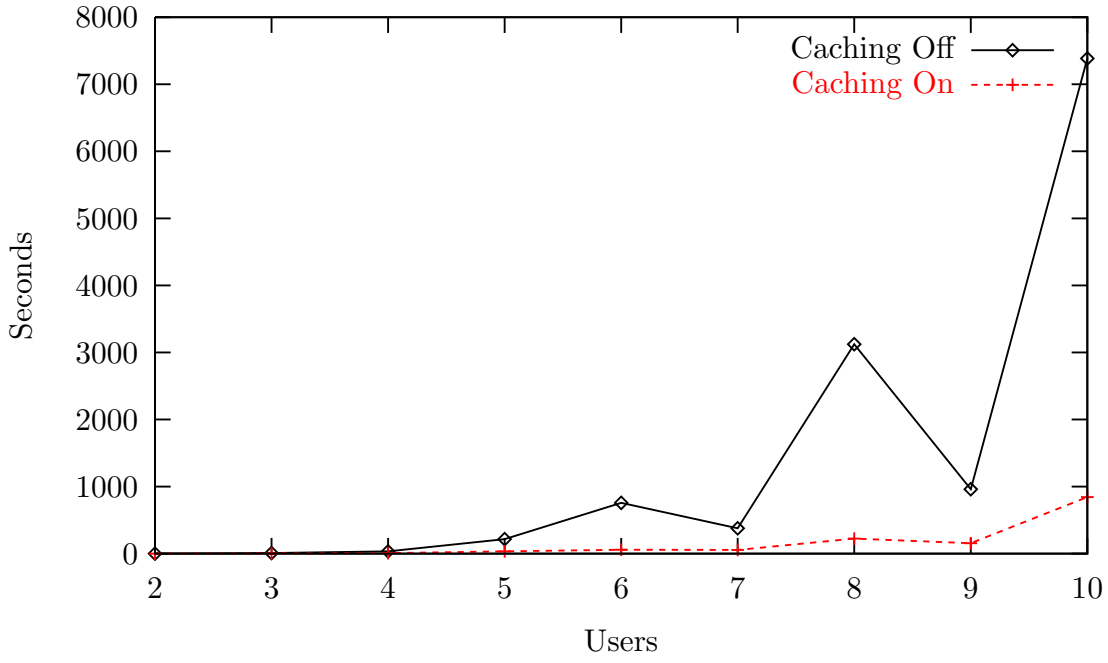


Figure 4.15: Execution time for greedy algorithm with bisection rate search against number of users with caching on and off

occur, resulting in a larger number of possible cache hits and in fact this behaviour is present in Figure 4.16.

In Figure 4.16 it is found that the correlation between number of executions and the absolute speedup begins to drop off at 10 ADSL users. This behaviour is observed because of the limitations of the cache size utilised in these experiments. By observing Figure 4.14 a drop in the cache hit:miss+collision ratio can be seen at 10 users. This is due to using a fixed bucket size (number of hash table entries) which starts to see a noticeable amount of cache collisions at 10 users. Using a large enough cache would mitigate this issue.

Figure 4.17 shows the total number of entries in the cache against the number of users. A best fit lines of aN^3 is also shown in Figure 4.17. It is believed that the number of entries in the cache should grow approximately proportional to N^3 as the number of bits loaded grows approximately in N and the number of bisection steps in N^2. If we assume that the best fit line represents the growth in cache size, then it follows that for 50 users the cache will contain approximately 10^9 entries.

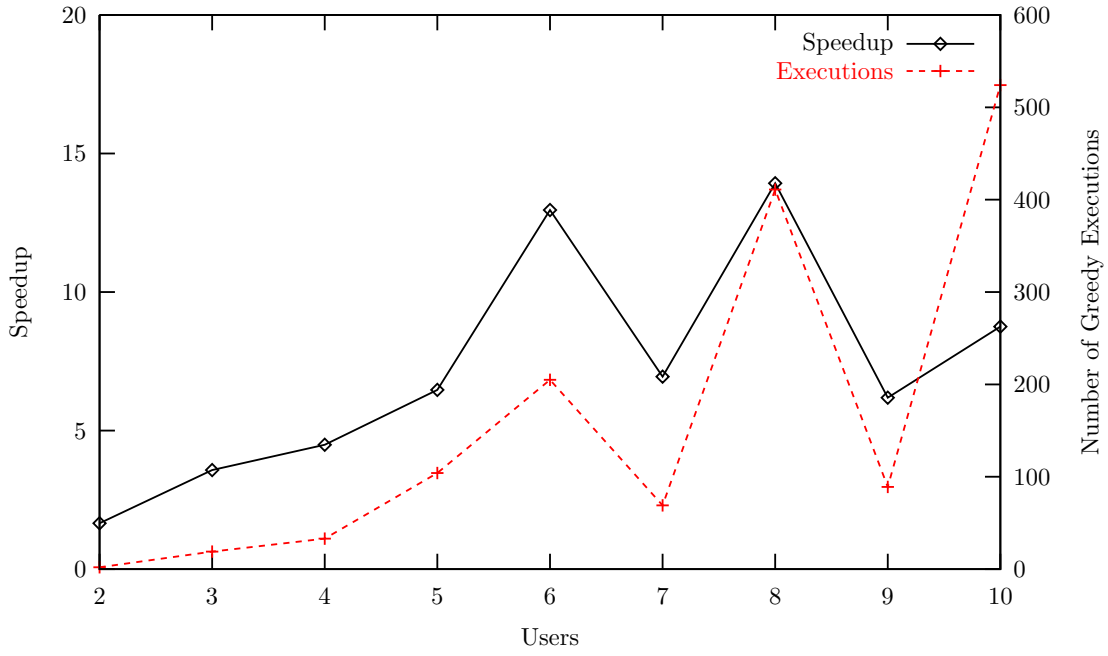


Figure 4.16: Speedup of PSD Vector caching over non PSD Vector caching greedy algorithm against number of users. Also shown is the number of executions of the greedy algorithm required

Hash	4 bytes
Bit Vector	50 * 4 bits = 25 bytes
PSD Vector	50 * 16 bits = 100 bytes

Table 4.4: Data structures present in the cache entry

Since the maximum power on any tone is $-36.5dBm/Hz(1mW)$, it is assumed that each power value can be adequately described in 10000 quantisation levels (much higher than the resolution required by the standard of 0.5dBm/Hz [47]) which equates to about 13.3 bits. Rounding up slightly, 2 bytes is used for each power value. Each bit value on each tone is in the range 0-15 and can therefore be stored in 4 bits. Using these values, the size of a cache entry for a 50 user network is approximately 129 bits as illustrated in Table 4.4.

Combining this value with the predicted number of cache entries gives an approximate figure of 130Gbytes for an ADSL2+ network.

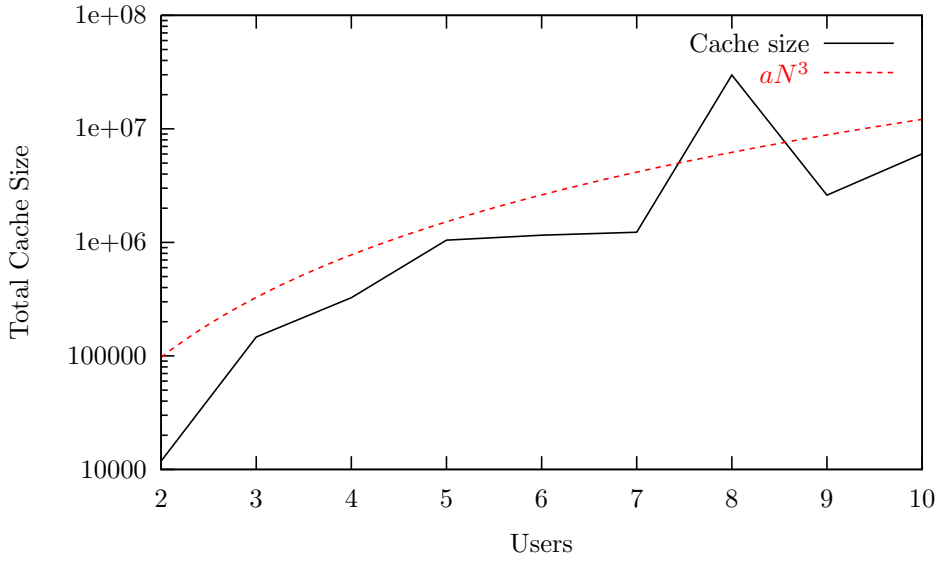


Figure 4.17: PSD Vector cache size against number of users for the greedy algorithm with rate bisection

4.4 Parallelisation

In this section, a new algorithm will be presented that enables parallelisation of the multiuser greedy algorithm rate search. By designing the algorithm to exploit the parallelism offered by modern processors, it is possible that the speed of the algorithm can be improved.

4.4.1 Parallel M-Section

This algorithm is a parallel version of the bisection algorithm which is hereafter known as parallel m-section. The basic idea is to divide the region of interest in the w search into smaller areas which are then evaluated in parallel. This reduces the convergence time for the per-user w search.

Figure 4.18 shows an illustration of a traditional bisection algorithm. The values of $w^0 \cdots w^3$ in Figure 4.18 represent the iterations of a bisection algorithm. Starting from an arbitrary initial value, the value w is reduced by a factor of 2 until a value of w which falls on the other side of the target rate is found. This value is now w^{min}. From this stage, the algorithm proceeds to traditional bisection and is shown to find a value with the required rate tolerance in one more iteration.

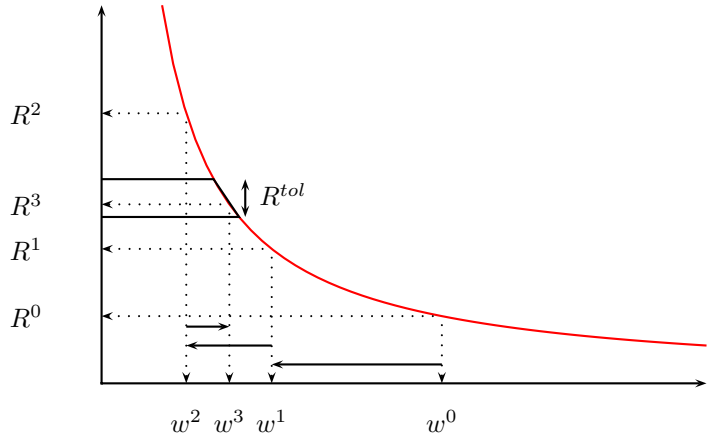


Figure 4.18: Illustration of a traditional bisection of the weight vector w

The illustration in Figure 4.18 is described algorithmically in Algorithm 15.

An illustration of two iterations of a parallel m-section with two threads is shown in Figure 4.19. At each iteration, the values of w tested are spaced equally between the values of w^{max} and w^{min}. It is assumed that values for w^{max} and w^{min} are already known, but this may not be the case in practise.

Having completed the first iteration in Figure 4.19 new values for w^{max} and w^{min} are found. This region is then divided up equally between the available threads and the process repeated until a value of w is found which meets the rate target on that line.

After each completed bisection of w, at least one of w^{min} or w^{max} is known for the next rate bisection by examining the results from the previous iteration. The only case where this is not true is on the very first bisection before the first execution of the greedy algorithm and in this case the initial values of w^{max} and w^{min} have to be selected. In the ordinary weight bisection algorithm the first value of w is always 1. In the parallel version with M threads, a method is required for selecting the M initial values of w.

In order to achieve this two functions are chosen to represent the initial values of w^{max} and w^{min}. It was found experimentally that in order to achieve better performance with increasing an increasing number of threads, the gaps between the initial w should decrease as the number of threads increases. This is achieved

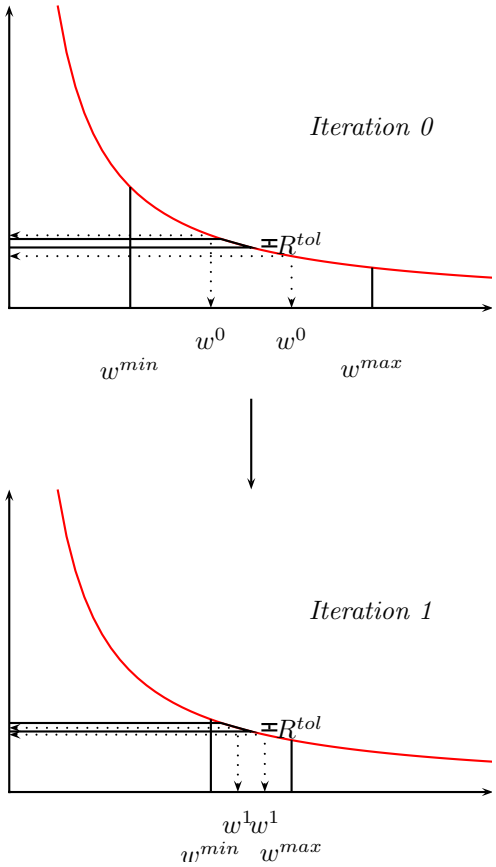


Figure 4.19: Illustration of a parallel m-section of the weight vector w with two threads

by choosing a lower function of:

$$w^{min} = \frac{1}{2^{\frac{M}{4}}} \tag{4.3}$$

and an upper function of:

$$w^{max} = \sqrt{M} \tag{4.4}$$

These functions are illustrated in Figure 4.20. Shown in Figure 4.20 are the initial w values chosen for each number of threads based on these upper and lower values.

The parallel m-section algorithm is shown in Algorithm 18 and illustrated in flow chart form in Figure 4.21.

Performance results for the parallel m-section algorithm are shown in figures 4.22 and 4.23. A significant performance increase is observed which flattens out

Algorithm 18 Parallel M-section Rate Search for Greedy Bit Loading

repeat
  for $n = 1 \dots N$ **do**
    if Iterations=0 **then**
      $w_{bottom} = \frac{1}{1^{\frac{M}{4}}}$
      $w_{top} = \sqrt{M}$
    else
      if $R_n^{last} > R_n^{target}$ **then**
        $w_{bottom} = w_n^{last}$
        $w_{top} = w_{bottom} * 2$
      else
        $w_{top} = w_n^{last}$
        $w_{bottom} = w_{top}/2$
      end if
    end if

    repeat
      if Iterations=0 **then**
        for $m = 1 \dots M$ **do**
          $w_n = w_{bottom} + \frac{w_{top} - w_{bottom}}{M+1} \times m$ ▷ See Note 1
          Execute Greedy Algorithm m ▷ In parallel
        end for
      else
        for $m = 1 \dots M$ **do**
          $w_n = w_{bottom} + \frac{w_{top} - w_{bottom}}{M-1} \times (m-1)$ ▷ See Note 2
          Execute Greedy Algorithm m ▷ In parallel
        end for
      end if
      $w_{bottom} = \text{argmin}_{w_n} R_n^m - R_n^{target}$
      $w_{top} = \text{argmin}_{w_n} R_n^{target} - R_n^m$
    until $|R_n - R_n^{target}| < R^{tol}$
  end for
until Rate Targets Met

Note 1: On first iteration use the starting values of w evenly spaced with $M+1$ gaps between w_{top} and w_{bottom}
Note 2: On subsequent iterations use starting values evenly spaced with M gaps between w_{top} and w_{bottom}

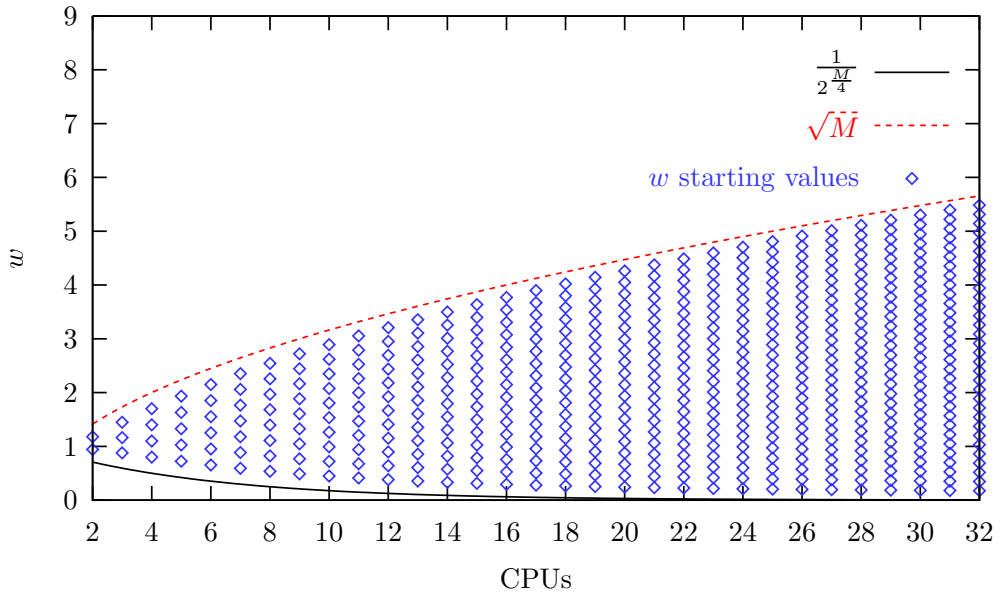


Figure 4.20: Functions for w_{max} and w_{min} plotted against total number of CPUs available

as the number of threads increases. The magnitude of the speedup also appears to increase with the number of users so we would expect to see an even larger speedup for most practical DSL scenarios.

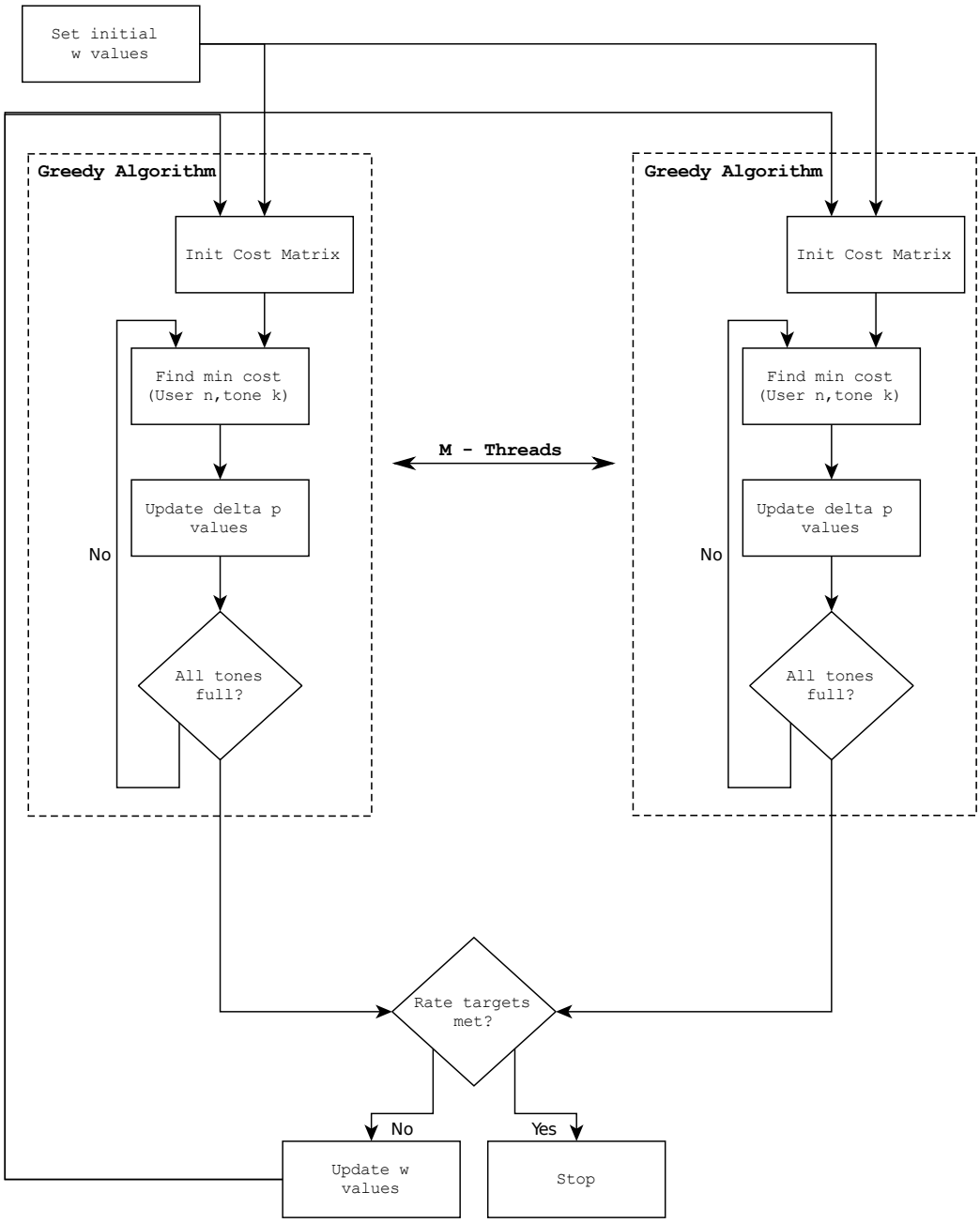


Figure 4.21: Flow diagram of parallel m-section algorithm

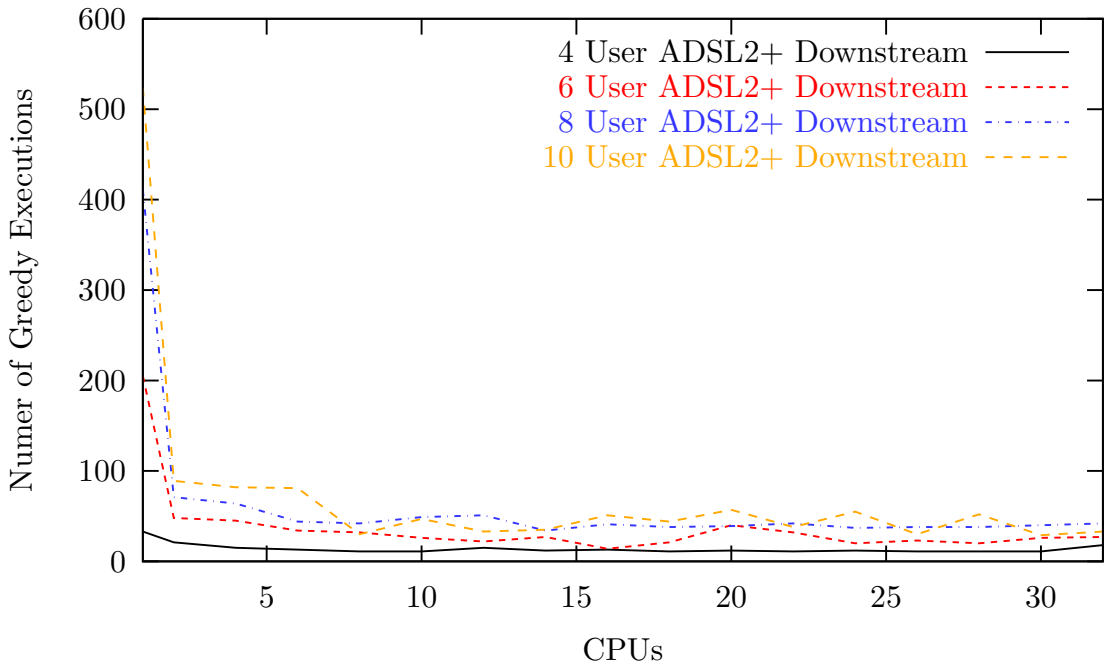


Figure 4.22: Number of greedy executions against number of CPUs for parallel M-section algorithm

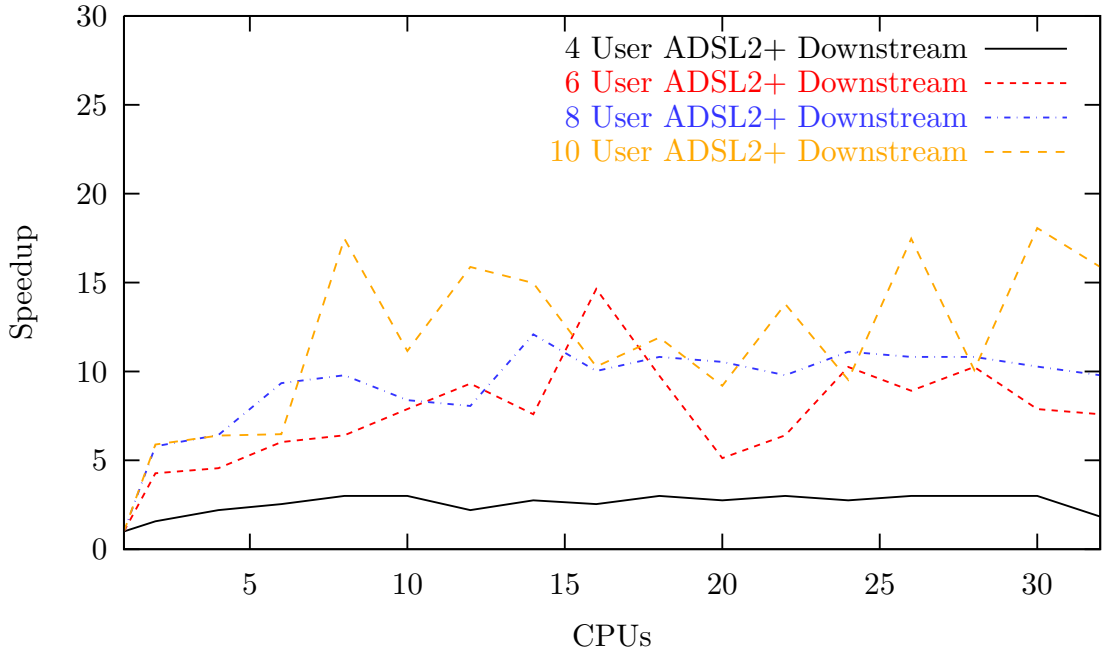


Figure 4.23: Speedup against number of CPUs for parallel M-section algorithm

4.5 Conclusions

In this chapter a number of enhancements to the Multiuser greedy algorithm were presented. By combining these techniques and high specification hardware, it is feasible that the greedy algorithm may be used for centralised DSM in the near future.

Using a bisection rate search, it was shown that the rate region performance of the greedy algorithm is close to the optimum and better than IWF and ISB for a range of DSL scenarios.

It was also shown that a sub-gradient rate search is possible and an adaptive sub-gradient rate search algorithm was presented to overcome the problem of convergence and step size selection. The adaptive sub-gradient was shown to be stable and exhibit a large performance advantage over bisection.

It was shown that a PSD cache could improve the performance of the greedy algorithm significantly. It is posited that a PSD cache for use in practical DSL scenarios (50+ lines), although very large, will be possible with high end hardware relatively soon. The PSD caching technique is also valid for other centralised DSM algorithms.

A parallel rate search algorithm was presented that can take advantage of multiple CPUs. This algorithm was shown to give large performance advantages over a single threaded bisection, but it probably not as efficient as using an adaptive sub-gradient approach.

Table 4.5 illustrates the execution times of various DSM algorithms which were obtained from simulation times on a Dell Inspiron 530n with a 2.33GHz Intel Core 2 E6550 with 2Gb of RAM. For the results including PSD caching, the cache parameters used were identical to those used for the results in Section 4.3. The DSL network configuration is taken from Figure 3.18 and the pattern is repeated for extra lines as necessary. The rate targets used are listed in Appendix A.1.

The data from Table 4.5 is visualised in Figure 4.24. The intractability of OSB, ISB and BBOSB are easily seen, which require over 1×10^5 seconds and

above for anything over 4 users. From this figure, the huge performance gains obtained using the various Greedy Algorithms are very apparent. For example, when using the Greedy Algorithm with an Adaptive Sub-Gradient rate search and PSD vector caching for a 7 User ADSL2+ scenario, a speedup of 3850 is obtained. The execution times in seconds in Table 4.5 do have not any particular significance individually but rather are used to compare the algorithm complexity on a reference platform.

Future work should seek to exploit parallelism in other parts of the algorithm to further decrease the execution time and make centralised DSM a realistic possibility.

Users	OSB	OSB + PSD Caching	ISB	ISB + PSD Caching	BBOSB	BBOSB + PSD Caching
2	35.65 s	14.6 s	32.21 s	2.73 s	7.12 s	2.88 s
3	3.24 hrs	2.12 hrs	16.22 mins	53.57 s	15.73 mins	4.2 minutes
4	†	*	3.93 hrs	9.35 mins	13.04 hrs	3.12 hours
5	*	*	12.06 hrs	30.04 mins	> 1 week	*
6	*	*	8.69 hrs	26.74 mins	*	*
7	*	*	28.46 hrs	1.41 hrs	*	*
8	*	*	*	†	*	*

* - not completed due to very long running time

† - could not be accurately calculated due to the large cache sizes causing swapping to disk on the simulation machine

Users	Greedy Bisection	Greedy Bisection + PSD Caching	Greedy Adaptive Sub Gradient + PSD Caching
2	0.26 s	0.199 s	0.78 s
3	9.99 s	3.89 s	5.46 s
4	35.6 s	10.96 s	10.29 s
5	3.71 mins	49.97 s	30.67 s
6	12.88 mins	1.7 mins	22.67 s
7	6.36 mins	1.21 mins	26.65 s
8	51.72 mins	5.67 mins	1.86 mins
9	15.92 mins	3.13 mins	1.41 mins
10	2.05 hours	13.14 mins	3.30 mins

Table 4.5: Execution times for various level 2 DSM algorithms using rate targets from Appendix A.1 and network configuration from Figure 3.18 repeated and the 1% worst case FEXT model

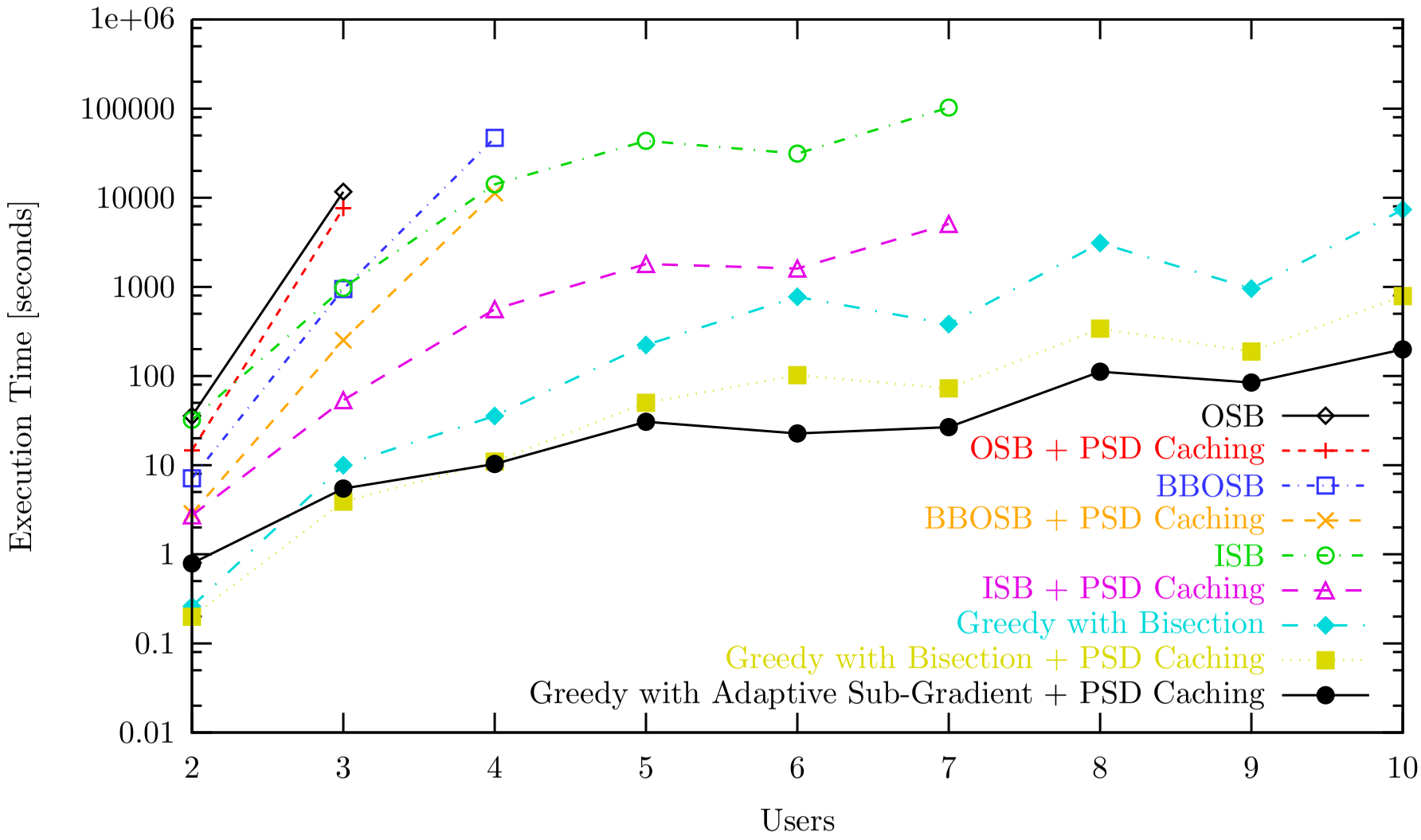


Figure 4.24: Illustration of DSM execution times from Table 4.5

Chapter 5

Conclusions

This thesis is primarily concerned with level 2 Dynamic Spectrum Management (DSM) algorithms for Digital Subscriber Lines (DSL). In Chapter 2, it was shown that currently available DSM techniques suffer from a variety of problems which make them problematic or unrealistic to utilise in today's DSL networks. In Chapters 3 and 4, a number of new algorithms for level 2 DSM were introduced, with the goal of making level 2 DSM practically realisable.

In Chapter 3, the development of a new level 2 DSM algorithm was detailed from its conception to design, known as Multi-User Incremental Power Balancing (MIPB). Later in this chapter, it was shown through extensive simulation results that MIPB produces excellent results comparable to the optimal result given by Optimal Spectrum Balancing (OSB) with equal line weights. It was also shown that the computational complexity of MIPB is significantly lower than that of other DSM level 2 algorithms and that it is tractable for large bundle sizes and was tested for up to 48 ADSL2+ lines. Up until now, no DSM level 2 results for more than 10 lines have appeared in literature. Parallelisation of MIPB was also investigated and it was discovered that excellent speedups could be obtained for up to 16 CPUs.

In Chapter 4, the Multi-User Greedy algorithm was revisited and it was found that with an adequate rate search algorithm, the Multi-User Greedy algorithm could produces near-optimal results in a range of DSL scenarios and outperform

other DSM algorithms such as Iterative Spectrum Balancing (ISB) and Iterative Water-Filling (IWF). A number of new rate search algorithms were developed, including one that can exploit parallel computation, and their relative performance assessed through simulation. A PSD vector caching technique was shown to vastly reduce the execution time of the greedy algorithm. To conclude this chapter, a summary of the absolute execution times for various algorithms was presented and it was shown that a very large reduction in computational complexity over other algorithms is obtained.

In comparison to ISB for 7 ADSL2+ users, the Greedy Algorithm with an Adaptive Sub-Gradient rate search and PSD vector caching was found to produce a solution 3850 times faster for a set of arbitrary rate targets.

Previous work on greedy algorithms which lead to the developments in this thesis were published in [15] and [16]. A patent application is currently underway based on the work in Chapters 3 and 4.

The work in this thesis goes some way to making practical level 2 DSM possible. With a fast enough level 2 DSM algorithm it is possible achieve:

- Data rate increases over IWF and SSM methods
- Power savings in the DSL bundle (i.e "Green-DSL")
- Rate adaptive DSM which further improves utilisation of the copper bundle by adapting to users bandwidth requirements

5.1 Future Work

The primary focus of this thesis has been the development of practical level 2 DSM algorithms. While there has been significant progress made in this area, there is still some distance to go before these algorithms can truly be executed in "real time". Although a parallel rate search algorithm was developed in this thesis, there is still much parallelism within the Multi-User Greedy algorithm which can be exploited to speed up the execution significantly. The obvious targets for this are parallelisation of the Δp update step which could be executed

on multiple cores in parallel. Also, the computations inside the Δp update step itself may be parallelised. Given that this step involves the solution of a large linear system in a large DSL bundle, this could be heavily parallelised and it an excellent candidate for execution on a GPU core.

If a "real time" level 2 DSM algorithm can be developed, it opens up the possibility of rate adaptive DSM to be implemented as illustrated in Section 1.2. In the rate regions show in Chapter 4, it can be seen how the rates of individual users can be traded off against one another. With a fast level 2 DSM algorithm in place, the rates of individual DSL lines could be traded against each other in response to the real time data rate requirements of each user, allowing faster peak rates as the applications demand it. In such a case, an algorithm which decides the real time rate requirements of each user would need to be developed.

References

[1] Thomas Starr, John M. Cioffi, and Peter J. Silverman. *Understanding Digital Subscriber Line Technology*, pages 67–80. Prentice Hall, 1999.

[2] P. Tsiaflakis, J. Vangorp, M. Moonen, and J. Verlinden. A low complexity branch and bound approach to optimal spectrum balancing for digital subscriber lines. In *GLOBECOM - IEEE Global Telecommunications Conference*, Nov. 2006.

[3] J. Lee, J.M. R.V. Sonalkar, and Cioffi. A multi-user power control algorithm for digital subscriber lines. *IEEE Communications Letters*, 9(3):193–195, March 2005.

[4] K. Stordahl. Broadband demand and the role of new technologies. *The 13th International Telecommunications Network Strategy and Planning Symposium*, 2008.

[5] J.M. Cioffi, H. Zou, A. Chowdhery, W. Lee, and S. Jagannathan. Greener copper with dynamic spectrum management. In *GLOBECOM - IEEE Global Telecommunications Conference*, pages 5697–5701, Dec. 2008.

[6] P. Tsiaflakis, Y. Yib, M. Chiangc, and M. Moonen. Green dsl: Energy-efficient dsm. In *IEEE International Conference on Communications*, Jun. 2009.

[7] T. Nordstrom, D. Statovci, and M. Wolkerstorfer. Energy efficient power back-off management for vdsl2 transmission. In *17th European Signal Processing Conference*, pages 2097–2101, Aug. 2009.

[8] Wei Yu, G. Ginis, and J.M. Cioffi. Distributed multiuser power control for digital subscriber lines. *IEEE Journal on Selected Areas in Communications*, 20(5):1105–1115, Jun. 2002.

[9] J. Huang, R. Cendrillon, M. Chiang, and M. Moonen. Autonomous spectrum balancing for frequency selective interference channels. pages 610–614, July 2006.

[10] W. Lee, Y. Kim, M.H. Brady, and J.M. Cioffi. Band-preference dynamic spectrum management in a dsl environment. *GLOBECOM - IEEE Global Telecommunications Conference*, pages 1–5, 2006.

[11] G. Ginis and J.M. Cioffi. Vectored transmission for digital subscriber line systems. *IEEE Journal on Selected Areas in Communications*, 20(5):1085–1104, Jun. 2002.

[12] F. Lindqvist, N. Lindqvist, B. Dortschy, P. Ödling, P.O. Börjesson, K. Ericson, and E. Pelaes. Crosstalk channel estimation via standardized two-port measurements. *EURASIP Journal on Advanced Signal Processing*, 2008:1–14, 2008.

[13] N. Papandreou and T. Antonakopoulos. Far-end crosstalk identification method based on channel training sequences. *IEEE Transactions on Instrumentation and Measurement*, 54(6):2204–2212, Dec. 2005.

[14] C.Zeng, C. Aldana, A.A Salvekar, and J.M. Cioffi. Crosstalk identification in xdsl systems. *IEEE Journal on Selected Areas in Communications*, 19(8):1488–1496, Aug. 2001.

[15] A. McKinley and A. Marshall. A new penalty based algorithm for multi-user spectrum balacing in xdsl networks. In *The 13th International Telecommunications Network Strategy and Planning Symposium, NETWORKS*, pages 1–23, Oct. 2008.

[16] A. McKinley and A. Marshall. Near-optimal multi-user greedy bit-loading for digital subscriber lines. In *Third International Conference on Access Networks*, pages 224–239, October 2008.

[17] R. Cendrillon, M. Moonen, J. Verlinden, T. Bostoen, and W. Yu. Optimal multiuser spectrum management for digital subscriber lines. In *IEEE International Conference on Communications*, volume 1, pages 1–5, 2004.

[18] Sumanth Jagannathan. *Interference and outage optimization in multi-user multi-carrier communication systems*. PhD thesis, Stanford University, 2008.

[19] Thomas Starr, John M. Cioffi, and Peter J. Silverman. *Understanding Digital Subscriber Line Technology*. Prentice Hall, 1999.

[20] P. Golden, H. Dedieu, and K. S. Jacobsen. *Fundamentals of DSL Technology*. Auerbach Publications, 2006.

[21] ANSI. Spectrum management for loop transmission systems. ANSI Standard T1.417-2003, September 2003.

[22] R. Baldemair, M. Horvat, and T. Nordstrom. Proposed method of crosstalk calculations in a distributed environment. ETSI ETC TM6 Plenary Meeting, 2003.

[23] Alcatel-Lucent, Adtran, and Conexant Systems Inc. Crosstalk channel modeling : detailed analysis and proposal. ANSI Contribution NIPP-NAI-2007-157R1, November 2007.

[24] AT&T, Adtran, and Conexant Systems Inc. 100x100 fext coupling matrix. ANSI Contribution NIPP-NAI-2007-010R2, November 2007.

[25] Alcatel-Lucent, Adtran, and Conexant Systems Inc. Revised 100 pair channel model with asymmetry. ANSI Contribution NIPP-NAI-2007-183, November 2007.

[26] J. A. C. Bingham. Multicarrier modulation for data transmission: an idea whose time has come. *IEEE Communications Magazine*, 28(5):5–14, May. 1990.

[27] J. M. Cioffi. A multicarrier primer. T1E1.4 contribution number 91-157, 1991.

[28] C.E. Shannon. Mathematical theory of communication. *Bell System Technical Journal*, 27(3-4):379–423, 1948.

[29] K.J. Kerpez. Near-end crosstalk is almost gaussian. *IEEE Transactions on Communications*, 41(5):670 –672, may 1993.

[30] F. Sjoberg, M. Isaksson, R. Nilsson, P. Odling, S.K. Wilson, and P.O. Borjesson. Zipper: a duplex method for vdsl based on dmt. *IEEE Transactions on Communications*, 47(8):1245–1252, Aug. 1999.

[31] C.M. Akujuobi, J. Shen, and M.N.O. Sadiku. A new parallel greedy bit-loading algorithm with fairness for multiple users in a dmt system. *IEEE Transactions on Communications*, 54(8):1374–1380, Aug. 2006.

[32] Thomas Starr, Peter Silverman, John Cioffi, and Massimo Sorbara. *DSL Advances*, pages 79–82. Prentice Hall, 2003.

[33] ATIS Committee. Dynamic spectrum management technical report. ATIS-PP-0600007, May 2007.

[34] R. Cendrillon and M. Moonen. Iterative spectrum balancing for digital subscriber lines. In *IEEE International Conference on Communications*, volume 3, pages 1937–1941, 2005.

[35] J. Papandriopoulos and J.S. Evans. Low-complexity distributed algorithms for spectrum balancing in multi-user dsl networks. *IEEE International Conference on Communications, ICC*, 7:3270–3275, Jun. 2006.

[36] Wei Yu and R. Lui. Dual methods for nonconvex spectrum optimization of multicarrier systems. *Communications, IEEE Transactions on*, 54(7):1310–1322, July 2006.

[37] Paschalis Tsiaflakis, Jan Vangorp, Marc Moonen, and Jan Verlinden. A low complexity optimal spectrum balancing algorithm for digital subscriber lines. *Signal Process.*, 87(7):1735–1753, 2007.

[38] Jungwon Lee, R.V. Sonalkar, and J.M. Cioffi. Multi-user discrete bit-loading for dmt-based dsl systems. In *GLOBECOM - IEEE Global Telecommunications Conference*, volume 2, pages 1259–1263 vol.2, Nov 2002.

[39] J. Lee, R.V Sonalkar, and J.M. Cioffi. Multiuser bit loading for multicarrier systems. *IEEE Transactions on Communications*, 54(7):1170–1174, July 2006.

[40] Raphael Cendrillon. *Multi-User Signal And Spectra Coordination For Digital Subscriber Lines*. PhD thesis, Katholieke Universiteit Leuven, 2004.

[41] Sumanth Jagannathan and John M. Cioffi. Distributed adaptive bit-loading for spectrum optimization in multi-user multicarrier systems. *Physical Communication*, 1(1):40 – 59, 2008.

[42] ITU. Very high speed digital subscriber line 2, itu-t recommendation g. 993.2. Technical report, ITU, 2006.

[43] V. Oksman and R. Stolle. Proposal for vdsl2 band-plan for profiles 17a and 30a. ANSI Contribution NIPP-NAI-2006-011R1, Jan. 2006.

[44] Thomas Starr, Peter Silverman, John Cioffi, and Massimo Sorbara. *DSL Advances*, pages 71–75. Prentice Hall, 2003.

[45] S. Graham, P. Kessler, and M. McKusick. gprof: a call graph execution profiler, 1982.

[46] P. Henkel. C++ threadpool framework built upon boost::thread. http://threadpool.sourceforge.net/.

[47] ITU. Physical layer management for digital subscriber line transcievers. ITU Std. G.997.1, 2003.

[48] M. Galassi et al. Gnu scientific library reference manual 3rd. edition, January 2009.

[49] G. Guennebaud and B. Jacob et al. Eigen2 library documentation v 2.0.5, 2009.

[50] A. McKinley and A. Marshall. An optimal multi-user, multi-service algorithm for dynamic spectrum management in dsl. In *Telecommunications, 2008. ICT 2008. International Conference on*, pages 1–5, June 2008.

Appendix A

A.1 ADSL2+ rate targets

This section contains tables of the rate targets used in bits per frame for the downstream ADSL2+ scenarios throughout the latter sections of Chapter 4. In each case, the target rate of line 1 is not set and is therefore maximised with all the the DSM algorithms tested.

Line 2	2000	8 Mbps

Rate targets for 2-User ADSL2+ downstream scenario

Line 2	2000	8 Mbps
Line 3	3200	12.8 Mbps

Rate targets for 3-User ADSL2+ downstream scenario

Line 2	2000	8 Mbps
Line 3	3200	12.8 Mbps
Line 4	4000	16 Mbps

Rate targets for 4-User ADSL2+ downstream scenario

Line 2	2000	8 Mbps
Line 3	3200	12.8 Mbps
Line 4	4000	16 Mbps
Line 5	3900	15.6 Mbps

Rate targets for 5-User ADSL2+ downstream scenario

Line 2	1800	7.2 Mbps
Line 3	3300	13.2 Mbps
Line 4	3800	15.2 Mbps
Line 5	5400	21.6 Mbps
Line 6	2600	10.4 Mbps

Rate targets for 6-User ADSL2+ downstream scenario

Line 2	1800	7.2 Mbps
Line 3	3300	13.2 Mbps
Line 4	3800	15.2 Mbps
Line 5	5400	21.6 Mbps
Line 6	2600	10.4 Mbps
Line 7	1600	6.4 Mbps

Rate targets for 7-User ADSL2+ downstream scenario

Line 2	1600	6.4 Mbps
Line 3	3200	12.8 Mbps
Line 4	3600	14.4 Mbps
Line 5	5200	20.8 Mbps
Line 6	2400	9.6 Mbps
Line 7	1400	6.4 Mbps
Line 8	1000	4 Mbps

Rate targets for 8-User ADSL2+ downstream scenario

Line 2	1600	6.4 Mbps
Line 3	3200	12.8 Mbps
Line 4	3600	14.4 Mbps
Line 5	5200	20.8 Mbps
Line 6	2400	9.6 Mbps
Line 7	1400	6.4 Mbps
Line 8	1000	4 Mbps
Line 9	2000	8 Mbps

Rate targets for 9-User ADSL2+ downstream scenario

Line 2	1600	6.4 Mbps
Line 3	3200	12.8 Mbps
Line 4	3600	14.4 Mbps
Line 5	5200	20.8 Mbps
Line 6	2400	9.6 Mbps
Line 7	1400	6.4 Mbps
Line 8	1000	4 Mbps
Line 9	2000	8 Mbps
Line 10	1500	6 Mbps

Rate targets for 10-User ADSL2+ downstream scenario

Appendix B

B.1 Architecture of DSL Bit-Loading Simulation

The DSM simulation tool used in this thesis was developed from scratch in C++. The channel models and crosstalk models utilised are detailed in sections 2.3 and 2.4. The GNU Scientific Library [48] was utilised for the solution of the linear system in equation 2.25 via LU decomposition.

During development the use of Eigen2 [49] (an advanced C++ linear algebra library with automatic vectorisation for SSE instructions) was also attempted. The Eigen2 library uses a partial pivoting LU decomposition which is faster but less numerically stable than that of a fully pivoting LU decomposition. It was discovered that the BBOSB algorithm would not function correctly using this library as it relies heavily on numerical stability of the upper and lower bound calculations.

The parallelisation of the DSM algorithms investigated was undertaken using posix threads and the *boost::threadpool* [46] library. The PSD vector cache uses a per tone lock (a pthread_mutex_t) to ensure that concurrent accesses from separate threads do not cause corruption of the cache.

The accuracy of the simulations were verified by comparing simulation results for OSB with a data set generated by Paschalis Tsiaflakis from Katholieke Universiteit Leuven.

Appendix C

C.1 List Of Publications

- A. McKinley and A. Marshall. An optimal multi-user, multi-service algorithm for dynamic spectrum management in dsl. In *Telecommunications, 2008. ICT 2008. International Conference on*, pages 1–5, June 2008

- A. McKinley and A. Marshall. Near-optimal multi-user greedy bit-loading for digital subscriber lines. In *Third International Conference on Access Networks*, pages 224–239, October 2008

- A. McKinley and A. Marshall. A new penalty based algorithm for multi-user spectrum balacing in xdsl networks. In *The 13th International Telecommunications Network Strategy and Planning Symposium, NETWORKS*, pages 1–23, Oct. 2008

- U.K. patent application - "Rate Adaptive Bit-Loading in DSL Bundles"