High-Speed bit-loading algorithms for Dynamic Spectrum Management in ADSL

Andrew Bolster

April 26, 2011

Executive Summary

Digital Subscriber Lines use advanced bit-loading algorithms to maximise spectral efficiency and throughput, often termed 'Dynamic Spectrum Magement" (DSM). Provider industry practise has been to use Level 1 DSM algorithms, where each line individually adjusts its power spectrum against the noise that it can sense. This is called "Waterfilling" and is not optimally efficient in terms of total bundle capacity.

Recent research into more advanced, Level 2, DSM algorithms, which consider the total noise characteristics of the bundle, show near-optimal performance, at a cost of significant computational complexity, making them unuseable in a consumer context.

The evolution of 'classical' vertex processing Graphic Processing Units (GPU's) into devices suitable for General Purpose computing on GPU (GPGPU) opens up these massively parallel floating point processors for practical computation, and introduces the possibility of re-implementing existing algorithms to leverage this new hardware, potentially allowing for near-optimal algorithms to be used in the field.

Acknowledgements

Thanks and praise must go to my project supervisor, Prof. Alan Marshall, whose occasional kick's-in-the-ass stopped me from going too far down the rabbit hole. Thanks also goes to Dr Alister McKinley, whose PhD work this project is based on, for his expert domain knowledge, clear thinking, and bar tab.

Originality Statement

'I hereby declare that this submission is my own work and to the best of my knowledge it contains no materials previously published or written by another person, or substantial proportions of material which have been accepted for the award of any other degree or diploma at Queen's University Belfast or any other educational institution, except where due acknowledgement is made in the thesis. Any contribution made to the research by others, with whom I have worked at QUB or elsewhere, is explicitly acknowledged in the thesis. I also declare that the intellectual content of this thesis is the product of my own work, except to the extent that assistance from others in the project's design and conception or in style, presentation and linguistic expression is acknowledged.'

Signed:_____

Contents

Appendices

Chapter 1

Introduction

If there are two characteristic the internet has shown over the past 30 years, it's spacial growth, and an insatiable demand for bandwidth. By December 2010, almost 30% of the world's population was connected to the Internet. While in the developing world, much of this can be accredited to the growth of mobile connectivity, but even for DSL, a technology that dates back to 1984 [], growth continues. In Q3 2010, BT announced that it had added nearly 200,000 DSL lines in the UK, bringing their total market share to 53%.**??** []

The rise of applications including live-video streaming such as BBC iPlayer, distributed content systems such as BitTorrent, and real-time gaming platforms such as XBox Live, continuously pressure Internet Service Providers (ISP's) to improve backbone and last-mile infrastructure to satisfy customer expectations. Technologies such as Fibre to the Home/Building/Cabinet (FTTx), which involves the replacement of copper lines with optical fibres, are coming to market,but in many cases, this is expensive and will not be done outside of dense municipal areas.

For other users, often Digital Subscriber Lines (DSL), built on top of the existing plain old telephone system (POTS) are the only option for anything approaching high-speed internet access. As such, DSL has been a focus of research and development, leading to significant growth in acheivable data rates, from 8MBits Downstream for Asymmetric DSL (ADSL) , through 24 MBits Downstream for ADSL2/2+, and more recently, up to 100MBits symmetrically on short loop lengths (¡ 250m) for Very high bitrate DSL (VDSL).

FTTx and xDSL are both evolving together; as FTTx reduces the last-mile length, xDSL frequency use expands to provide the maximum bandwidth on that short-loop. While in some urban areas, FTTH will become the prevalent form of broadband internet access, xDSL will

still be needed for a long stime to come for the vast majority of the populace.

But existing methods of manageing the bandwidth allocations for bundles of xDSL lines are either computationally simple, but sub-optimal, or computationally intractable and near-optimal. Much work and research has gone into developing advanced and intelligent Dynamic Spectrum Management (DSM) algorithms, but little attention has been applied to the practical implications of these algorithms and the real-world effect that different forms of that implementation make to performance and tractability.

Massively parallel computing, in the form of GPGPU, is becoming a popular paradigm in computational research1, with classically linear algorithms being distributed across these Single Instruction Multiple Data (SIMD) devices giving general speedups in the range of 5-500x. The most popular and developed form of GPGPU architecture is nVidia's Compute Unified Device Architecture (CUDA), which leverages the streaming multiprocessor (SM) and high speed memories or their GPU's (usually used for gaming) for use as scientific computation devices.

../images/GPGPU-search-growth.png

Figure 1: Growth in Google Scholar papers mentioning GPGPU

In this report, the viability of GPGPU accelerated DSM algorithms is investigated,

1.1 Project Specification

School of Electrical and Electronic Engineering and Computer Science
FINAL YEAR PROJECT 2010/2011

Title:	High-Speed bit-loading algorithms for Dynamic Spectrum Management in ADSL
Student:	Andrew Bolster
Supervisor:	Prof A Marshall
Moderator:	Dr J McAllister
Area:	Digital Comms
Research Cluster:	Digital Comms

Digital subscriber lines need to employ bit-loading techniques to improve their throughput; this isknown as "Dynamic Spectrum Management" (DSM). Currently all implementations of DSM use level 1whereby individual lines use a "waterfilling" algorithm to adjust the power allocated to each tone in thespectrum, and a central management station adjusts the waterfilling parameters for each line in thesubscriber bundle. However this approach is sub-optimal whenever the capacity of the overall bundleis considered, hence more recent research has proposed (level 2) more intelligent methods toallocate the power for each bit in each tone, considering both near and far end crosstalk in the bundle. A major problem with the level 2 techniques is their computational complexity which currently rendersthem practically infeasible, e.g. ISB typically takes ¿ 1 week to compute the tones for a 10-linebundle.Graphic Processing Units (GPUs) represent a new approach to massively parallel floating-point computations. Moreover the Compute Unified Device Architecture (CUDA) developed by NVidia,represents a framework whereby new highly parallel algorithms can be developed. The main aim ofthis project is to apply this approach to level 2 DSM algorithms, many of which are highly parallel intheir operation. **The objectives of the project are:**

1. Become familiar with DSM techniques for Digital Subscriber Lines.
2. Become familiar with the CUDA environment for GPU's and identify a suitable platform.
3. Investigate efficient implementations of level 2 DSM.
4. Develop an implementation of a level 2 bit-loading algorithm using GPU's.

M.Eng. Extensions

1. Analyse the performance of your implementation in terms of speed, cost, and scalability

(number of lines).

2. Compare your design with existing implementations.

Learning Outcomes

1. Understand how to use CUDA to programme GPU's.
2. Be able to design bit-loading algorithms for DMT.
3. How to analyse the performance of an implementation.

1.2 Overview and Objectives

The project specification is quite expansive, covering a vast range of areas that I have not encountered directly in my studies but have tangentially met externally. Parallel computation is an active area of research, especially Massively Parallel GPGPU systems such as this. Additionally, there are almost as many DSM algorithms as there are DSM researchers, so algorithm selection and implementation must be very carefully planned and executed to keep within a workable timeframe.

As with any project of this scale, the necessary first stage is a thorough investigation of the problem. Not only the areas of DSM directly relevent to the Level 2 algorithms under analysis, but also the underlying DSL technology to enable generation and implementation of a computational model on which to test the algorithms. This will necessitate the selection of a programming environment that must be held consisitent throughout the project, as well as the additional implementation of Optimal Spectrum Balancing (OSB) for comparative analysis. In the interests of not reinventing the wheel, research will also be undertaken to assess the usefulness of any existing DSL models / DSM implementations in terms of development guidance.

Beyond the problem itself, technologies involved in the suggested solution will have to be expansively researched, such as CUDA, GPGPU, and Computational Parallelism, including any recent relevent scholarly works that could be useful.

To summarise, the main points that need investigation will be:

- Problem Research
    - How DSL works

- – DMT communications
- – Dynamic Spectrum Management
- – DSM Levels
- – Existing/previous DSL simulation systems
- – Existing/previous DSM implementations

- Solution Research

  - – Relevant Programming Environments
  - – Software Profiling
  - – Massively Parallel Computing
  - – Parallel Computing Architectures
  - – CUDA software considerations
  - – CUDA hardware considerations

- Development

  - – Design and implement DSL system model simulator
  - – Implement Optimal Spectrum Balancing (OSB) algorithm
  - – Create an analysis server
  - – Evaluate system performance

Chapter 2

Research & Background

The modern Broadband telecommunications network has augmented the already massive effect on society of The Internet, allowing access to information from all over the world to one's doorstep. Broadband rollout and specifically xDSL has spurred the growth of VoIP technologies, Video chat and streaming services, and always-on connectivity, fundamentally changing how society interacts; in 2010, over 31 million adults in the UK made purchases online, that's 62% of the adult population buying from Amazon, Tesco Direct, and eBay instead of going to brick-and-mortar retailers. http://www.statistics.gov.uk/cci/nugget.asp?id=8

Almost 30% of the UK have access to 'Broadband' networks; defined as having a downstream bandwidth of 144Mbits, triple the number from just five years previously. Nielsen reports that in 2010, over 82% of the population had some form of internet access available to them; this number up from just under 60% over the same period.

In terms of speed, while variability is rife within the British Isles, Ofcom ... http://stakeholders.ofcom.org.uk/m data-research/telecoms-research/broadband-speeds/speeds-nov-dec-2010/

While this is due to a variety of technologies, xDSL plays a massive role; as FTTx technologies increase the available backhaul bandwidth, xDSL technologies are still needed for the ever-shortening (and ever faster) local subscriber loop. Indeed, it is forecast that by 2012, only 4% of Broadband connections with be complete FTTx lines from subscriber to provider. DSL is still alive and well, and this paper will show that there is still plenty of capacity in existing lines that, if used efficiently, can keep up with subscriber demand for at least the next five years.

2.1 DSL

Originally part of the 1984 ISDN specification, but like most communications technologies, draws its basis from Claud Shannon's 1948 work at Bell Labs.[1]. DSL operates by overlaying wideband digitally modulated data signals on top of existing baseband voice signals on the POTS; over the phone line. This overlayed signal does not effect the voice line, as DSL models modulate on frequencies from around 4kHz to as high as 4MHz; well above the 300-3400Hz baseband service¡ref¿¡/ref¿, so both operations can be one asynchronously and simultaneously with little to no interference from each other[1]

2.1.1 DSL Modulation and Signal Transmission

Orthogonal Frequency-Division Multiplexing (OFDM), or as it is standardised within DSL[3], Discrete Multi-Tone (DMT), is the most common modulation scheme in xDSL, where by a large number of orthogonal (non-interfering) sub-channels, tightly packed in the frequency domain, are used to carry data. Across these numerous sub-channels, data is effectively carried in parallel, and stream data is encoded and split up across these sub-channels at a relatively low symbol rate[4]. In the case of xDSL, this is generally in the range of 2-15 bits per sub-channel[2].

ODFM is used in a wide variety of wideband communications systems, including 802.11a/g/n, WiMax, and DVB-T/H aswell as xDSL, and provides many advantages to service providers;

1. Tolerant of Multi-path effects such as fading, and Intersymbol Interference (ISI)

2. Tolerant of narrow-band co-channel interference.

3. Comparatively high spectral efficiency against spread spectrum modulation systems.

4. Efficiently implemented using Fast Fourier Transform (FFT)

5. Run-time adaptable to severe channel conditions without advanced equalisation techniques.

But it is to be noted that OFDM suffers from some disadvantages;

---

[1]the POTS service can affect the performance of a DSL service if the phone lines are not low-pass filtered and split, but this is now common practise[2]

7

1. Doppler shift sensitivity

2. Highly sensitive to synchronisation issues

3. Poor power efficiency due to a High Peak to average power ratio (PAPR) necessitating linear transmission circuitry

¡ref¿¡/ref¿ "'need to work out how to cite this section as adapted from wikipedia"'

The ITU G.992 specifications, or G.DMT are the ITU specified sub-schemas of OFDM that are used in Asynchronous DSL deployments (ADSL, ADSL2, and ADSL2+), and ITU G.993 specifies the newer Very-high-bitrate DSL technologies (VSDL, VSDL2).

Other xDSL systems such as HDSL, IDSL, RADSL, SDSL, and SHDSL use a combination of Carrierless Amplitude Phase (CAP) and Two Binary One Quaternary (2B1Q) modulation schemes, which are not the subject of this report, but are interesting to investigate none the less.

Carrierless Amplitude Phase modulation was the de facto scheme for ADSL up until around 1996[5], and was almost completely abandoned for ADSL after the 1999 ratification of ADSL ITU G.992.1 Interoperability standard, but lives on in Rate Adaptive DSL (RADSL), and some HDSL and SDSL deployments.

Two Binary One Quaternary modulation, as the name indicates, uses four signalling levels to represent a two bit, Gray coded input signal, meaning that if a signal was misread by one-level, only one bit error would occur. In xDSL, 2B1Q is used in ISDN over DSL (IDSL), and some flavours of HDSL and SDSL.

As stated, the DMT system as implemented in most A/VDSL systems, operates across hundreds of sub-channels, or slices of the frequency spectrum. How the streaming input data is multiplexed, quantised, and modulated across these sub-channels is largely variable. This Spectrum Management problem will be dealt with in more detail later in this report, but for now, it is suffice to say that data can be dynamically spread across and away from 'poor', noisy subchannels, and also can be concentrated on 'good', clean areas of the spectrum. This technique is called bit-loading, and entails the use of bit-variable modulation, usually based around variable constellation Quadrature Amplitude Modulation, and is demonstrated in the two diagrams **??**.

The capacity of a single DMT line is well established, coming again from Shannon's seminal work[1]. His 'ideal' solution, 'Water-filling', simply assigns bit-loads based on the SNR inverse

of the line; pouring energy into sub-channels with the best SNR, i.e. most capable of carrying the signal. As the 'best' sub-channels are filled, the ajoining, less capable channels are filled, and so on until no more energy can be added to the channel.

As stated, this is and ideal approach, and presents the maximum theoretical capacity of the line. Practical implications of this, or any other bitloading algorithm, must sacrifice 'ideality' for realiability. As such, they must implement what is called a 'Shannon Gap' or 'SNR Gap', a sub-ideal limit on the power loaded on a channel to maintain reliable communications. This sub ject is revisited in ¡INSERT REFERENCE HERE¿, but the time being, any M-QAM sub-channel constellation, the number of bits that can be encoded within is given in 2.1 [1].

$$b = log_2(M) \tag{2.1}$$

But before going too deeply into the 'problem', first an exploration of the landscape within which the problem lies.

2.1.2   DSL System Architecture

A classical DSL local loop architecture involves a Central Office (CO), with many Line Termination (LT) modems, each connected to a Network Termination (NT) modem at the Customer Premises (CP). Each LT/NT pair has a separate line between them, and for local loops, these lines are wrapped together in bundles of up to fifty individual lines. This bundle then runs from the CO to the CP's, with intermediate CP lines 'peeling off' the bundle. Additionally, along the length of the bundle, additional lines can 'peel in' to the bundle. Some of the wide variety of possible situations are shown diagrammatically in ¡ref¿¡/ref¿.

These bundles make for a highly noisy RF environment; with signals from each line interfering with every other line in the bundle through electromagnetic induction of leaked signal power. This phenomenon is termed crosstalk and is the major limiting factor on the achievable data rates of DSL rollouts, with no current practical solution.

Crosstalk reduces the SNR on DMT sub-channels on a 'victim' line, reducing the amount of bits-per-symbol that can be transmitted on that sub-channel at a pre-defined constant error rate. The level of SNR reduction is often in the 10-20dB range and as such, is the most dominant noise source experienced within the bundle. [6]

This phenomenon comes in two major forms;

9

- Near-End Crosstalk (NEXT): signal leakage from LT-LT or NT-NT

- Far-End Crosstalk (FEXT): signal leakage from LT-NT or LT-NT

In the vast majority of current DSL rollouts, NEXT is effectively eliminated through the use of Frequency Division Duplexing; i.e the upstream data (NT-¿LT) is sent on a very different frequency than the downstream (LT-¿NT), and as such does not cause direct interference with a 'neighbouring' termination point.

The amount of crosstalk (XT) is generally time-constant (excluding severe temperature and humidity changes) and as such can be computationally 'accomodated for' a-priori, but direct vectorisation (we'll find out about that later) and even in-direct cross talk elimination is computationally difficult, if not intractable.

Before one can understand the effects of XT on a DSL system, and especially before one can experiment with different cross talk 'avoidance' algorithms, a realistic dynamic simulation of a functional DSL system must be generated.

2.1.2.1 DSL System Modelling

Modelling a DSL system, instead of opting for 'real-world' values, allows experimentation and observation of channel behaviours that may not be immediatly clear from end-point experimentation [2]

Simulation in this case involves the generation of a cross-talk gain (XTG) matrix for an N-line network for each of its K operating channels. This XTG matrix will then be used by the management algorithms to assess the 'cost' or indeed viability of a particular bitload on a particular line on a particular channel.

As such it is important to understand the electromagnetic transmission characteristics of twisted pair phone lines. This area is largely concerned with the generation of per-line-per-channel transfer functions for interacting sections within a bundle. It is shown in [7] that for standard (Category 3) lines, the following RCLG characterisation is stable up to the 30MHz area, at which point this 'simplified' characterisation veers away from real-world performance due to high-frequency envelope shearing. Category 5 lines are stable with this kind of charactarisation up to around 150MHz.

---

[2]not to mention that real-world experimentation is time-consuming, expensive, and often in-comperable to other results due to a multitude of measurement standards

RLCG Characterisation is derived from the per-unit-length two-port model shown in **??**, which can be viewed as a infinitisemally small section of a segment of transmission line. The RCLG parameters represent Resistance, Inductance, Capacitance and Conductance per unit length of line. The direct gain channel models used in this report are based on the long line approximations taken from [8], which in turn were taken from [7].

This assumption is that (for a long DSL line) the input line impedance V_1 matches the characteristic impedance V_2. Given that assumption aswell as;

- d :line length

- Z :Impedance per unit length:$R \cdot L$

- Y :Admittance per unit length: $G \cdot C$

- Z_l :load impedance

- Z_s :source impedance

- Z_0 :characteristic impedance:$\sqrt{\frac{Z}{Y}}$

- γ :propagation constant:$\sqrt{Z \cdot Y}$

the transfer function for a given DSL line is;

$$H = \frac{Z_0 \cdot \mathsf{sech}(\gamma d)}{Z_s \cdot [\frac{Z_0}{Z_1} + \mathsf{tanh}(\gamma d)] + Z_0 \cdot [1 + \frac{Z_0}{Z_1} \cdot \mathsf{tanh}(\gamma d)]} \tag{2.2}$$

To ensure that this modelled transfer function is smooth with respect to frequency [3], the RLCG values are parameterised thusly;

$$R(f) = \frac{1}{\frac{1}{\sqrt[4]{r_{0c}^4 + a_c \cdot f^2}} + \frac{1}{\sqrt[4]{r_{0s}^4 + a_s \cdot f^2}}} \tag{2.3}$$

Where r_{0x} is the DC resistance of copper (c) and steel (s) and a_x are skin effect related constants

$$L(f) = \frac{l_0 + l_\infty (\frac{f}{f_m})^b}{1 + (\frac{f}{f_b})^b} \tag{2.4}$$

---

[3]necessary due to the large margins of error in characterising 'real-world' lines

Where l_x are low (0) and high (∞) frequency inductance and b and f_m define the transition from low to high frequencies.

$$C(f) = c_\infty + c_0 \cdot f^{-c_e} \tag{2.5}$$

Where C_x represent contact (∞) and other $(0, e)$ capacitances, chosen from measurements.

$$G(f) = g_0 \cdot f^{+g_e} \tag{2.6}$$

Where G_x represent $(0, e)$ conductances, chosen from measurements.

Within this project, as in[8], AWG 24 line parameters were used, shown in table **??**.

Using these parameters, **??** shows the gain-curve for a variety of lengths of AWG24 cabling produced from (2.2)

2.1.2.2   Crosstalk Modelling

The above model allows the generation of 'idealised gains' for a given line, ignoring all external effects, such as the afor mentioned crosstalk. To bring crosstalk into the picture, assuming that NEXT is eliminated completely by FDD, we can model FEXT based on the industry standard ETSI 1%[9] model(2.7)

$$|H_{FEXT}(f)|^2 = N^{0.06} K_{FEXT} f^2 |H_{channel}(f, L)|^2 \tag{2.7}$$

The "1%" in this case represents the worst-case-scenario, and in a real deployment would not be encountered 99% of the time. This is partially driven from systemic pessamism, but partially comes from the fact that this worst-case scenario allows for a modelling function that is smooth with frequency.

In most cases, this 1% model is over zealous in its estimations of crosstalk coupling, and also ignores spatiality within the bundle [4]. Using real data from a four sub-bundle, 100x100 DSL binder, the existing model can be modified to more closely match the coupling data of the 'real' bundle.

---

[4]i.e., lines in the core of the bundle recieve must more crosstalk than those on the skin of the bundle

$$|H_{FEXT}^{j,k}(f)|^2 = |H_{FEXT}^{j,k}(f)|^2 \times 10^{\frac{X_{dB}}{10}} \qquad (2.8)$$

Where the modifier X is selected based on a Beta probability distribution of sample data, for example,[10].

As stated, DSL is a crosstalk-limited system, and as such, the effect of crosstalkers also effects some of the fundamental assumptions that must be made about the transmission characteristics of the lines in the bundle.

From[11], the approximate Shannon Gap for a DSL system based on M-QAM is derived thus. From the Central Limit Theorem, as the number of Crosstalkers increases, the effects of those crosstalkers approximates to a Gaussian Distribution[12]. Therefore, a DSL channel can be approximated as a an Additive White Gaussian Noise (AWGN) channel, which is described[1]:

$$C(k) = log_2(1 + SNR(k)) \qquad (2.9)$$

This represents the maximum possible capacity (in bits) on a given channel ($C(k)$), ignoring practical considerations such as coding complexity and processing delays. From[1] and[4], the probability of a symbol error for an uncoded M-QAM constellation [5] on a given sub-channel is shown in (2.10)

$$P_e \approx N_e Q\left(\sqrt{\frac{3}{M-1}SNR}\right) \qquad (2.10)$$

Where $Q(x)$ is the probability of unitary Gaussian variable will will exceed x, given in (2.11). In general, DSL systems aim for this probability of symbol error (P_e) to be around $1e^{-7}$.

$$Q(x) = \int_x^\infty \frac{1}{\sqrt{2\pi}} e^{-u^2/2} du \qquad (2.11)$$

(2.11) is generally rewritten in terms of the standard error function $erf(x)$, shown in (2.12) and (2.13).

$$Q(x) = \frac{1}{2}\left(1 - erf(\frac{x}{\sqrt{2}})\right) \qquad (2.12)$$

---

[5]with an associated number of nearest neighbours, N_e

$$Q^{-1}(x) = \sqrt{2} erf^{-1}(1 - 2x) \tag{2.13}$$

Rearranging (2.10) for M gives (2.14)

$$M = 1 + \frac{SNR}{\frac{1}{3}(Q^{-1}(\frac{P_e}{N_e})^2)} \tag{2.14}$$

Revisiting (2.1), (2.15) can be obtained, expressing the number of bits that can be encoded on a sub-channel.

$$b = log_2\left(1 + \frac{SNR}{\frac{1}{3}(Q^{-1}(\frac{P_e}{N_e})^2)}\right) \tag{2.15}$$

Contrasting (2.15) with (2.9), the uncoded channel gap is defined in (2.16).

$$\Gamma_{uncoded} = \frac{1}{3}\left(Q^{-1}(\frac{P_e}{N_e})\right)^2 \tag{2.16}$$

This allows simplification of (2.15) with respect to (2.16) into (2.17)

$$b = log_2\left(1 + \frac{SNR}{\Gamma_{uncoded}}\right) \tag{2.17}$$

This characterisation of the Shannon Gap is not quite complete, and represents the best-case scenario within a DSL system, while ignoring potential coding gains from Forward Error Correction such as the Trellis[13] or Reed-Solomon[14] coding schemes.

As such, two additional modifiers are added to the Γ calculation; a performance margin γ_m which allows for SNR 'headroom' to maintain error rates during temporarily bad noise conditions, and a coding gain γ_{eg} which incorporates any error correction modulation in place. This gives a final Γ sum shown in (2.18)

$$\Gamma = \Gamma_{uncoded} + \gamma_m + \gamma_{eg} \tag{2.18}$$

In[8] it is demonstrated that crosstalk coupling can be assumed not to have any practically relevant effect on sub-channels other than the sub-channel from which that crosstalk origi-

nates, such as when DMT blocks are transmitted and recieved synchronously or when cyclic prefixing and pulse shaping ("Zipper" DMT[15]) is used.

From[16], (2.19) shows the maximally optimal bit-loading on line n of N users, and tone k of K total sub-channels, including the above derivation for FEXT coupling.

$$b_n(k) = log_2 \left(1 + \frac{p_n(k)|h_{nn}(k)|^2}{\Gamma \left(\sigma_n^2(k) + \sum_{j \neq n}^{N} p_j(k)|h_{jn}(k)|^2 \right)} \right) \tag{2.19}$$

In (2.19), the following definitions are provided for clarity;

- $h_{ij}(k)$:The crosstalk gain from user i to user j on channel k

- $p_n(k)$:The power provisioned on channel k for user n

- $\sigma_n^2(k)$:The background noise power experienced by user n on channel k

- Γ :The Shannon Gap derived from (2.18)

By letting $f(b_n(k)) = \Gamma(2^{b_n(k)} - 1)$, (2.19) can be rearranged to (2.20)

$$p_n(k) - f(b_n(k)) \sum_{j \neq n}^{N} p_j(k) \frac{|h_{jn}(k)|^2}{|h_{nn}(k)|^2} = f(b_n(k)) \frac{\sigma_n^2(k)}{|h_{nn}(k)|^2} \tag{2.20}$$

It is clear that (2.20) can be characterised as an N-dimensional linear system of equations, of the form $A(k)P(k) = B(k)$, where

$$A(k)_{ij} = \begin{cases} 1, & \text{for } i = j \\ \frac{-f(b_i(k))|h_{ji}|^2}{|h_{ii}|^2}, & \text{for } i \neq j \end{cases} \tag{2.21}$$

$$P(k) = [p_1(k) \dots p_i(k) \dots p_N(k)]^T \tag{2.22}$$

$$B(k) = \left[\frac{f(b_1(k))\sigma_1^2}{|h_{11}|^2} \dots \frac{f(b_i(k))\sigma_i^2}{|h_{ii}|^2} \dots \frac{f(b_N(k))\sigma_N^2}{|h_{NN}|^2} \right]^T \tag{2.23}$$

As such, the vector $B(k)$, as a function of $b(k$, describes the amount of bits loaded onto each user's line on tone k, and $P(k)$ perscribes the power required on each line to support the

vector $B(k)$ of bits loaded onto those lines on channel k. It is the solution and optimisation of this system of equations that is the fundamental limiting factor, and drive for, DSM systems.

The reasoning for this is primarily visible looking at the concept of rate-regions, i.e an N-dimensional surface of possible maximum line bit-loads for each user n. **??** shows the relationship between two given users, R_1 and R_2) within a given bundle. It is evident that one <u>could</u> achieve an very high data-rate on one line, but at great expense to the other. Mathematically, the search for optimal bitloading is summed up in (2.24).

$$\max_{\{p_n \in P_n\}_n} \sum_n w_n R_n \qquad (2.24)$$

Using an optimal rate-region, of the style demonstrated in **??** can be generated algorithmically from the above equations. But often these algorithms can be computationally intractable, and less computationally expensive (but sub-optimal) algorithms are used instead, producing reduced rate regions, as shown, which do not take full advantage of the availiable spectrum given the same crosstalk and bundle noise characteristics.

In essence, this Spectrum Management Problem, as stated in (2.24), can be expanded based on line-weighting; where some lines get 'preferential treatment', either because they are paying more, or as part of a lead-and-lag rate-compensation system[6]. Thus, the Spectrum Management Problem can be generalised to N users as a maximisation of a weighted sum of data rates within a bundle, as shown in (2.25).

$$\sum_{n=1}^{N} w_n \sum_{k=1}^{K} log_2 \left(1 + \frac{p_n(k_| h_{nn}(k)|^2}{\Gamma\left(\sigma_n^2(k) + \sum_{m \neq n}^{N} p_m(k)|h_{jm}(k)|^2\right)} \right) \qquad (2.25)$$
$$s.t. \sum_{k=1}^{K} p_n(k) \leq P_{max,n} \forall n$$

The optimisation of this problem in a dynamic near-realtime timeframe is the major focus of DSM research, which is summarised next.

---

[6]Lag and Lead, terms originally from the field of economics but 'adopted' by the field of control theory, implies a kind of 'lending' system, where by a given variable (in this case the bit-rate/weight on a particular line) can be allowed to 'run rampant' if the system has spare capacity, but that added usage is accounted and is 'paid back' to the system when needed, and often that particular variable will be more 'borrowed from' until its lag/lead accounts are even again

2.1.3 Dynamic Spectrum Management Introduction

A general solution to this rate-region problem has been the use of Static Spectrum Management (SSM), where each line has a pre-defined power spectral density (PSD) profile under which it operates. This is sometimes called Spectral Masking. Unfortunately this masking is done once (or a few times, if lines are added to / removed from the bundle), and additionally assumes that each line will be fully utilised at all times. This a major abstraction from reality, and has driven the development of spectrum management systems that can dynamically re-allocate PSD's based on near-realtime usage. From[17]:

> Suppose that 25 copper lines exist in a cable bundle. Depending on the time (day versus night, weekday versus weekend) and the location (commercial versus residential area), the demand for DSL services varies. For examplle, the data traffic load at an office is heavy during the daytime and may be close to zero at night. Furthermore, depending on the specific loop topologies, the interference caused by one line to another varies. Some pairs might be physically close in the binder over a very short distance; hence, the interference between these pairs might be very small. On the other hand, other pairs might be proximate along most of the loop length, which leads to a very strong interference between those two pairs ... Revisiting the example where 25 copper lines exist in a telephone bundle, suppose that in the middle of the night at school, one PhD student is trying to download a large paper from a website using DSL. If static spectrum management is used, the speed is as slow at night as during the day, when many other students are also using their DSLs. However, if the DSL network uses [Dynamic Spectrum Management], the speed at night can be much faster than during the daytime because the bitrate on the line in use can be optimised to take advantage of the low-noise conditions.

It is clear that with the addition of some usage-based intelligence, the bundle could be much more effectively used. Dynamic Spectrum Management systems allow for power and rate allocations for each line to be dynamically controlled, but are classified into four general levels, based on the amount of coordination between lines, ranging from completely autonomuous operation[7] to centralised signal vectoring and control, as summarised in 2.1.3, adapted from[18] and[8].

---

[7]Strictly speaking not DSM but included for completeness

DSM Level	Description	Examples
0	No DSM, Completely Autonymous per-modem Spectrum Management	NA
1	Single line spectrum balancing with politeness and signal impulse control	IWF[8],ASB[9]
2	Multiple line spectrum balancing with spectra controls	OSB[10], ISB[11],SCALE[12]
3	Multiple line signal coordination (vectoring)	Vectored-DMT[19], SVD[13]

This coordination (where applicable) is controlled centrally from a Spectrum Maintenance Center (SMC), which recieves data from Customer Premisis Equipment (CPE) via an Auto-Configuration Server (ACS), and/or from DSLAMs and LT-side equipment via an Element Management System (EMS). The SMC then collates the recieved information, and distributes profile settings for the DSLs.

2.1.4   DSM Level 0/1

For clarity, levels 0 and 1 are generally grouped, as the only major difference is that while in level0[14], equipment is completly autonomous in terms of its own spectrum management; level 1 systems recieve power and rate limitations from the SMC, within which they must operate.

Thinking back to the initial discussion of DMT spectrum balancing, the most common technique for autonomous spectrum assignments is based on Water-filling[1], and refined for DSL in[22] [?]

For the single user case, refer back to **??**, but in the multi-user case, water-filling attempts must be iteratively refined (Hence, Iterative Water-Filling, or IWF); so as to allow the 'other' lines to see each lines new noise profile, see **??** for a graphical example of this. This helps to reduce the effects of the so-called "Near Far Problem", whereby the recieved signal from crosstalker which is closer to the reciever, is stronger than its direct line transmitter signal.

---

[8]Iterative WaterFilling
[9]Autonomous Spectrum Balancing
[10]Optimal Spectrum Balancing
[11]Iterative Spectrum Balancing
[12]Successive Convex Approximation for Low ComplExity[20]
[13]Single Vector Decomposition[21]
[14]currently the most common method of 'Spectrum Balancing'

The waterfilling algorithm produces the optimal solution[15], maximising channel capacity for a single user with a continuous bit-assignment [16]. In practise this is incorrect, as integer bits must be assigned to each channel for transmittion. When this integer condition is applied, it is known as "Discrete Waterfilling". Discrete waterfilling has two variations; one which focuses on maximising bitrate with respect to a stated power budget, known as Rate-Adaptive (RA); and another which focuses on minimising the power requirements with respect to a stated bitrate, known as Fixed-Margin (FM). These two conditions are mathematically phrased in (2.26) (2.27)[7].

$$
\begin{aligned}
maxR &= \sum_{k=1}^{K} b(k) \\
s.t. &\quad \sum_{k=1}^{K} p(k) \leq P_{budget}
\end{aligned}
\tag{2.26}
$$

$$
\begin{aligned}
min &\quad \sum_{k=1}^{K} p(k) \\
s.t. &\quad \sum_{k=1}^{K} b(k) \geq B_{budget}
\end{aligned}
\tag{2.27}
$$

The resolution of both of these forms of waterfilling is surprisingly simple in theory; find the tone with the lowest bit-addition energy [17], and add a bit to it. This continues until either the power or rate budget is reached, depending on the mode applied. This basic algorithm is the Levin-Campello (LC) Algorithm[24].

2.1.4.1   Iterative Water Filling

One of the first forms of DSM, IWF is computationally simple, de-centralised, and relatively easy to implement. The rational of IWF is to limit iteratively perform LC bit-loading and to adapt the power constraints so as to limit the power used in the bundle while maintaining total bundle capacity, thereby lowering the power of crosstalking signals and actually increasing the net bundle data rate greatly when compared to SSM, but is far from optimal, and is not guaranteed to converge, or settle, on on any result. The general algorithm for IWF is shown in 2.1.4.1

repeat
    for n=1...N **do**

---

[15]It is proven in[23] that for frequencies less than 2MHz, there exists only one Nash Equilibrium point within a distributed DSL network, and although it is possible, no example of multiple optimal rate-region points has been discovered to date[17]

[16]I.e non-integer bit-assignments are permitted

[17]generally termed $\Delta p(k)$

Execute LC Algorithm with power budget P_n on line n

if $R_n > R_n^{target}$ **then**

$\quad P_n = P_n - \gamma$

else

$\quad P_n = P_n + \gamma$

end if

end for

until convergence

2.1.5 DSM Level 2

2.1.5.1 Optimal Spectrum Balancing

From[6], OSB optimally solves the Spectrum Management Problem for multiple users, but is highly computationally intensive and generally cannot be practically computed for more than four lines[8]. The avenue taken by OSB in solving the Spectrum Management Problem is a mathematically complex one. The solution is centered around the use of a Lagrangian dual decomposition of the initially states Spectrum Management Problem (2.25). To explain what this process entails, one must look closer at the problem; from[6]

> The fundamental problem is that the total power constraints on the modems couple the optimisation across frequency. As such optimisation must be done jointly across all tones, which leads to an exponential complexity in K. We overcome this problem through the use of the dual decomposition method. This technique allows us to replace the constrained optimisation problem with an unconstrained maximization of a Lagrangian[18] The Lagrangian incorporated the constraints implicitly into the cost function, removing the need for the constraints to be explicitly enforces. As a result, the optimisation canbe decoupled across frequency, and an optimal solution can be found in a per-tone fashion. This leads to a linear rather than exponential complexity in K and a computationally tractable problem.

---

[18]A Lagrangian of a dynamical system is a function that summarises the dynamics of the system, and in the field of optimisation, allows for the joint-optimisation of a dually (in this case) constrained problem through the use of an additional Lagrange multiplier to generate the new Lagrangian decomposition L that implicitly encompasses both (all) constraint functions

In practical terms, that means that the Spectrum Management Problem can be 'simplified' to include the pan-channel power constraints, meaning that instead of generating many solutions to the general (first line of (2.25)) problem, and then subsequently discarding those as they do not satisfy the bundle power constraints, global power is a focal consideration. The Lagrangian decomposition of (2.25), encompassing the power constraints, is shown in (2.28)

$$L = \sum_{n=1}^{N} w_n \sum_{k=1}^{K} log_2 \left(1 + \frac{p_n(k_|h_{nn}(k)|^2}{\Gamma \left(\sigma_n^2(k) + \sum_{m \neq n}^{N} p_m(k)|h_{jm}(k)|^2 \right)} \right) - \lambda_n \left(\sum_{k=1}^{K} p_n(k) \right) \quad (2.28)$$

Is was shown by Yu and Lui[25] that this decomposition is an exact match to the originally stated problem for large numbers of channels.

It is also stated in[8] that (2.28) can be tonally decoupled, and a (unfortunately non-convex) Lagrangian expression for each channel generated, as in (2.29), and by optimising for a maximal per-tone Lagrangian, and searching λ space[19], the original problem can be solved.

$$L(k) = \sum_{n=1}^{N} w_n b_n(k) - \sum_{n=1}^{N} \lambda_n p_n(k) \quad (2.29)$$

Since (2.29) is non-convex, an exhaustive search across b_n space is required[20] Even with the Lagrangian decomposition, the computationally explosive nature of this style of exhaustive bit-field search renders OSB computationally intractable for more than four or five lines. Some runtimes from[8] are shown in **??**

The generalised algorithm for OSB is shown in 2.1.5.1

repeat
  repeat
    for $k = 1 \dots K$ **do**
      $\arg\max_{b_(k)} L(k) = \sum_{n=1}^{N} w_n b_n(k) - \sum_{n=1}^{N} \lambda_n p_n(k)$

---

[19]How the λ space is searched is a matter of much debate, which will be met later

[20]For the interested reader, this is a computationally explosive procedure; consider the vector $B(k)$, with each vector index as the bit load on a particular line on channel k, and a maximum bits per tone of 15 (from the DSL standard). For two lines, this represents only 225 possibilities, but for four, its 50625; for 8 its over 2.5 billion, and for a standard 50 line bundle, its a 59 digit number. To put that in perspective, the number of stars in the observable universe is only a 24 digit number. For a given max bits per tone b_{max}, the number of bit combinations for N lines is b_{max}^N

      Solve by N-d exhaustive search

   end for

   $\lambda_n = \lambda_n + \epsilon(\sum_{k=1}^{K} p_n(k) - P_n^{max})$

 until λ convergence

 $w_n = w_n + \epsilon(\sum_{k=1}^{K} b_n(k) - R_n^{target})$

until w convergence

OSB can be augmented using a Branch and Bound searching structure into Branch and Bound OSB (BBOSB). BBOSB is covered in detail in[26], but suffice to say, even with the improved searching structure, BBOSB is still exponential in N, but with a lower complexity coefficient; where OSB is only tractable for 4-5 lines, BBOSB is tractable up to approximately 10 lines[8].

2.1.5.2   Iterative Spectrum Balancing

ISB is based on the same Lagrangian decomposition as OSB, but instead of an exhaustive search across the bit-space within the Lagrangian, each line in turn is optimised, i.e a linear search instead of OSB's matrix-search. It was presented[27] by Cendrillon and Moonen in 2005 and thoroughly investigated by Yu and Lui[25] in 2006, and is near-optimal, but is not guaranteed to converge on a global maximum.[27].

The improvement in computational complexity is great; OSB has a N complexity of $O(2^N)$, whereas ISB attains a complexity of $O(N^2)$,meaning that for practical bundle sizes, ISB is computationally tractable. The ISB algorithm is shown in 2.1.5.2

repeat

  for $k = 1 \ldots K$ **do**

    repeat

      for $n = 1 \ldots N$ **do**

        Fix $b_m(k) \forall m \neq n$

        $\arg\max_{b_{(k)}} L(k) = \sum_{n=1}^{N} w_n b_n(k) - \sum_{n=1}^{N} \lambda_n p_n(k)$

        $\lambda_n = \lambda_n + \epsilon(\sum_{k=1}^{K} p_n(k) - P_n^{max})$

        Solve by 1-d exhaustive search

      end for

    until λ convergence

  end for

  $w_n = w_n + \epsilon(\sum_{k=1}^{K} b_n(k) - R_n^{target})$

until w convergence

2.1.5.3   Multi-User Greedy Spectrum Balancing

It has been shown that while the LC algorithm is optimal in the case of a single line, the straightforward multi-user expansion of this (IWF) is decidedly sub-optimal. Cioffi, Sonalkar, and Lee, in[28] presented a heuristic extension to the Levin-Campello Algorithm, termed Multi-User Greedy Spectrum Balancing. The heuristic applied was, instead of in IWF where each line is individually optimised, the bundle is viewed as a whole, and bits are iteratively assigned to both the line and channel with the minimum cost of addition. This cost of bit-incrementing, which for IWF was simply $\Delta p_m(k)$ where m was the line being balanced, and k was the channel on the line with the lowest additional power requirement; additionally includes the sum of $\Delta p_n(k)$ on all other lines required to accomodate the additional crosstalk generated on the incremented line2.30

$$C(m,k) = \sum_{n=1}^{N} \left(\underset{p_n(k)}{b(k)+1} - \underset{p_n(k)}{b(k)} \right) \tag{2.30}$$

2.1.5.4   Multi-User Incremental Power Balancing

In[8], McKinley details the development of MIPB as coming from a critical analysis of Multi-User Greedy Spectrum Balancing (MUGSB). One of the major deficiencies of MUGSB is the (significant) possibility of the algorithm getting stuck in local minima, or getting 'deadlocked', whereby bits cannot be added to any lines due to any other line in the bundle attempting to violate its power constraint to accomodate the additional cross talk incurred, (see 2.30). To remedy this, an adaptive power penalty function is used to stop 'clean' lines being continuously loaded up to their power budget, and then 'locking' all other lines, and to instead force all the lines to be more or less equalised as the algorithm progresses. The algorithm for MIPB is shown in 2.1.5.4. The final bundle efficiency of MIPB, while non-optimal, is very close, but the real improvement is in runtime performance; 10 lines with rate targeting applied in under five minutes.

repeat
    FIXME
until REALLY, FIXME!

2.1.6 DSM Level 3

Level 3 techniques rely on the concept of signal vectoring, coordinated at a DSLAM, such that all lines are terminated on common equipment. Complete signal vectoring eliminates crosstalk through the generation of a cancellation matrix consisting of block-permuted, diagonal Inverse Discrete Fourier Transform matrices, acting as a Generalised Decision Feedback Equaliser, which, as demonstrated in[29], can eliminate crosstalk, aswell ISI in the case of impulse envelope skewing over long lines.

Vectored-DMT (VDMT) can produce optimal bundle efficiencies, but the conditions on VDMT's operation[21] make it inapplicable for commercial use.

Fortunately, DSL Level 3 is not the subject of this project.

2.2 Parallel Computing

Today, most (hopefully all) engineering and computer science students are educated in programming, in one form or another. The current languages of choice;C,C++, Java, and Python, all (classically) conform to what is known as procedural, or imperative, programming, where by instructions are written to be serially fed into a processor, and data flow is dictated by run-time conditional redirection. Ever since the computing innovations of Turning in the 1940's[30] and the later codification of the Von Neumann Architecture (VNA) after his work on EDVAC [22][31], the concept of a serially operated computer dominated the field of computer science for decades.

Parallel computing, on the other hand, which simultaneously uses many processing units to solve a problem by breaking a problem (or data) set up and distributing this split workload to different processing units[23], has been a niche interest. Parallel computing was for a long time the reserve of a few highly specialised machines created over the years, generally in the fields of medicine, energy or warfare.[24]

---

[21]Co-location of at least one side of the bundle, computationally expensive pre-computation, inflexibility of applications, requirement for expensive real-time DSP hardware at both bundle terminations

[22]Electronic Discrete Variable Automatic Computer, prodominantly used for Ordance Trajectory calculations and other military applications

[23]Whether instruction pipelining should be included in this is a point of debate, which will be dealt with later in this chapter

[24]For example, the Cray-1 was built for fusion plasma dynamics research, the Cytocomputer pipelined processor was built for biomedical image processing in 1980[32], and the Staran Content-Addressable Parallel

The reason for this 'edging out' of parallel computation was simple; Moore's Law[34], which perscribes that computational density doubles more-or-less every 18 months, held for about four decades. This continuous, explosive, growth drove programmers with computationally intense problems simply to wait 18 months for their applications to run twice as fast on newer hardware, instead of using problem decomposition to solve the current problem in a more distributed way. This period of exponential processing growth in respect to hardware has, in recent years, come up against major blocks; quantum physics[25], the infamous 'power wall'[26], and a general consumer and industrial drive towards low power devices (including 'supercomputers').

Since the early 2000's, the semiconductor industry has settled on two main paths to stave-off the demise of Moore's Law (in its current form[27]); multi-core and many-core.

Multi-core processing involved the use of a relatively small number of monolithic processing units, very akin to traditional single CPU's, placed on a single die, maintaining the execution speed of existing sequential programs while allowing for direct execution concurrency. The latest 'state of the art' in this field is the Intel i7 Sandy Bridge 32nm architecture, with 8 processing cores[37], each of which is a hyperthreaded[28] x86 instruction set processor, giving, in theory, 16 hardware threads of parallelism. The trend in these types of devices has actually almost matched Moore's Law in its simplest form; the number of cores in multi-core chipsets has been roughly doubling every two years.

Many-core computing, on the other hand, involves the use of a relatively large number of 'dumb' cores. While many many-core architectures exist[29] the area in which parallel computing research and applications has been recently most focused is the development and adaptation of consumer graphics hardware to application acceleration and scientific computing, termed General Purpose computing on GPU's (GPGPU).

While it is not directly relevant to this document, outside of the semiconductor industry, the drive towards massively disturbed clusters of not-necessarily-co-located machines [30] has become such a major feature of the scientific computing landscape, it is cureently being used at

---

computer, built in 1972, was used for Cartographic data processing for military intelligence[33]

[25]Quantum Tunnelling leads to non-deterministic behaviour at around the 16 nanometer mark[35]

[26]If processor speeds were growing today at the same rate as in the mid-90's, CPU's would require a power density equivalent to that of a Saturn V booster stage nozzel[36]

[27]Moore himself has said the 'his' law is the exception to Murphy's Law, in that no-matter what doomsday predictions are made regarding it, The Law seems to perpetuate itself in new forms

[28]Read:two hardware threads per core

[29]IBM's BlueGene/Q architecture, Intel's Larabee Microarchitecture, AMD's FireStream, NVIDIA's Tesla, and Tilera's iMesh to name a few

[30]Made famous by SETI@Home in 1999

CERN. Data processing for CERN's LHC project is handled by a worldwide grid of over 200,000 processing cores, with 150 PB of storage, and a theoretical capacity to handle the 27TB of raw data per day coming out of the LHC, communicating across a mixture of dedicated fibre connections and the public internet backbone. On a smaller scale, programming paradigms such as Message Passing, and Parallel Virtual Machines, allow applications to be executed against an arbitrary number of computing nodes in a cluster.

Before looking back atat GPGPU in detail, it is important to establish the 'families' of parallelism; namely 'Flynn's Taxonomy'

2.2.1 Forms Of Parallelism, aka, Flynns Taxonomy

Based on the interactions between instructions and data, Flynn[38] classified computation systems in terms of how Processing Units (PU)[31] process instructions and data.

Class	Description	Example
SISD	Single Instruction, Single Data Stream	Traditional Serial Processing
SIMD	Single Instruction, Multiple Data Stream	GPGPU[32]/Many-core
MIST	Multiple Instruction, Single Data Stream	(Rare[33])
MIMD	Multiple Instruction, Multiple Data Stream	Cluster/Grid systems (eg. LHC-Grid)

Intuiatively, this breakdown is a matrix of data and instruction parallelism; Taking SISD and SIMD separately (as they are the most familiar and relevant), SISD performs a single operation on a single piece of data. It can only process that one piece of data at a time, and can only execute one instruction at a time[34]. SIMD on the other hand takes that one execution instruction, and can operate on many pieces of data, producing many results in parallel. This behaviour is augmented in MIMD, where data and instructions are processed in parallel; this could be within a single device (Such as the Cell Broadband Engine) or distributed across the globe.

Flynn created a very clear and simple map of parallelism. Unfortunately the real world got in the way and complicated things. Two major 'complications' will be addressed that have

---

[31]Think of cores or threads of execution

[32]Not really anymore, but we're getting to that

[33]This architecture is generally only applied where redundancy is critical, and the 'Multiple Instructions' operate on the signel datastream and must agree. One prominent example is the Space Shuttle Flight Control systems

[34]Don't worry dear readers, pipelining is coming...
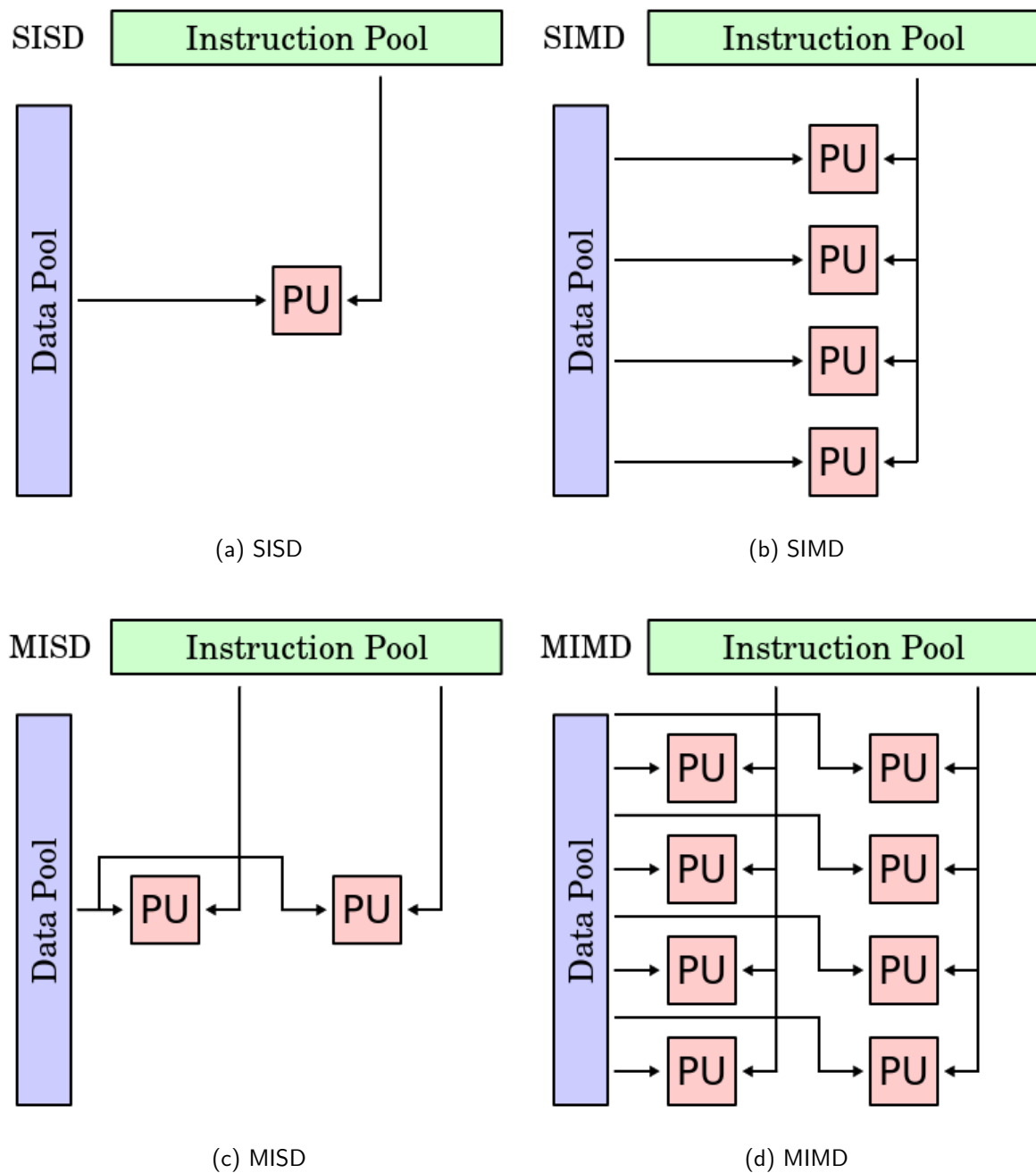
(a) SISD

(b) SIMD

(c) MISD

(d) MIMD

Figure 2: Flynns Taxonomy of Parallel Computing Systems

muddied the taxonomic water; SISD Instruction Pipelining, and Single Program Multiple Data (SPMD) execution.

2.2.1.1 Pipelining

Computer processors are not monolithic boxes that have data pumped into them and the results instantly come out; within even a basic microprocessor, there are a variety of hardware sections that work in sequence, making data and instructions flow through the device to produce the correct result. This sequence is termed the 'pipeline'.

A Generic pipeline consists of four steps;

1. Fetch

2. Decode

3. Execute

4. Write-back

Simply put, Fetch grabs the next instruction to be executed from local memory, Decode processes that instruction, prepares the processor's Arithmetic Logic Unit (ALU) for execution, aswell as establishing appropriate data memory channels. During Execute, the calculation is performed, and during Write-back, the result of the calculation is stored back in memory. For the sake of simplicity it can be assumed that each of these stages takes one clock cycle. These tasks are performed by different areas of the chip. This opened up the possibility of having multiple instruction pipelines 'running' at once;

Looking back to Flynn's Taxonomy, it appears at first glance that what was previously a solid SISD processor could be considered MIMD; the data fetched can be different for each pipelined instruction, and those instructions do not have to be the same either. Whether this counts as true parallelism or not is not as important as its performance improvement; where previously one instruction took four clock cycles to execute, with this simple four stage pipeline, four instructions can be processed in 8 clock cycles.[35]

2.2.1.2 Single Program Multiple Data

SISD implementations, at least as were developed in when Flynn stated his Taxonomy, meant that each machine-level instruction was executed in parallel in lock-step. This meant that data-dependent conditional execution simply wasn't possible without suspending the operation of

---

[35]Complications to this system arise if the instructions processed are conditional, but as pipelining is not the focus of this project, refer to [39] for more information
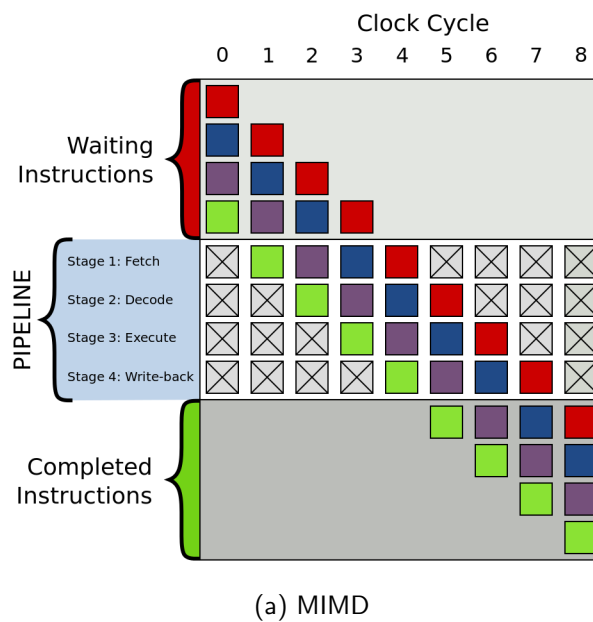
(a) MIMD

Figure 3: Example of a 4 Stage Pipeline system

PUs for which that condition was not satisfied. SPMD is a subset of MIMD whereby each PU can continue on at its own execution pace, regardless of the state of other PUs. This is exemplified in the Messate Passing Interface (MPI) model, where by a (usually) identical program is executed by all PUs, and to control collective program flow and distribution of data, messages are send and recieved asynchronously between PUs. These PUs could be processor cores within a single machine, or indeed on the same, multi-threading, core, or they could be on machines on opposite sides of the globe.

2.2.2   CUDA Execution architecture

2.2.3   CUDA Memory architecture

References

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

[2] Peter J. Silverman Thomas Starr, John M. Cioffi. DSL Advances. Prentice Hall, 2003.

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

[4] J. M. Cioffi. A multicarrier primer. ANSI T1E1, 1991.

[5] Burton R. Saltzberg. Wiley Encyclopedia of Telecommunications, chapter Carrierless Amplitude Phase Modulation. John Wiley & Sons, Inc, 2003.

[6] J. Verlinden R. Cendrillon, M. Moonen. Optimal multiuser spectrum management for digital subscriber lines. IEEE International Conference on Communications, 1:1–5, 2004.

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

[8] Alastair McKinley. Novel Bit-Loading Algorithms For Spectrum Balancing In Digital Subscriber Lines. PhD thesis, Queen's University Belfast, September 2009.

[9] ANSI. Spectrum management for loop transmission systems. ANSI, t1.417-2003 edition, September 2003.

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

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

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

[13] G. Ungerboeck. Channel coding with multilevel/phase signals. IEEE Transactions on Information Theory, IT-28:55–67, 1982.

[14] I. S. Reed and G. Solomon. Polynomial Codes over Certain Finite Fields. Journal of the Society for Industrial and Applied Mathematics, 8:300–304, January 1959.

[15] R. Nilsson F. Sjoberg, M. Isaksson and jesson. Zipper: a duplex method for vdsl based on dmt. IEEE Transactions on Communications,, 54(8):1245–1252, August 1999.

[16] M.N.O. C.M. Akujuobi, J. Shen 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, August 2006.

[17] Seong Taek Chung. Implementation and Applications of DSL Technology, chapter Dynamic Spectrum Management. Auerback Publications, 2007.

[18] V. Pourahmad J. Cioffi, M. Brady. Vectored DSLs with DSM: The road to Ubiquitous Gigabit DSLs, 2006.

[19] J.M. Cioffi G. Ginis. Vectored-DMT: A FEXT cancelling DSL System. Globecom 2000, 2000.

[20] J.S. Papandriopoulos, J.; Evans. Low-Complexity Distributed Algorithms for Spectrum Balancing in Multi-User DSL Networks. IEEE International Conference on Communications, 2006. ICC '06., pages 3270–3275, June 2006.

[21] Georg Taubock and Werner Henkel. MIMO Systems in the Subscriber-Line Network. In 5th International OFDM-Workshop 2000, 2000.

[22] Wei Yu and J.M. Cioffi. Competitive equilibrium in the Gaussian interference channel. IEEE International Symposium on Information Theory, 2000. Proceedings., 2000.

[23] Jungwon Lee Seong Taek Chung, Seung Jean Kim. A game-theoretic approach to power allocation in frequency-selective gaussian interference channels. IEEE International Symposium on Information Theory, 2003. Proceedings., June 2003.

[24] H.Levin. A Complete and Optimal Data Allocation Method for Practical DMT systems. IEE JSAC, April 2001.

[25] Wei Yu and R. Lui. Dual methods for nonconvex spectrum optimization of multicarrier systems. Communications, IEEE Transactions on,, June 2006.

[26] M. Moonen P. Tsiaflakis, J. Vangorp. A low complex- ity branch and bound approach to optimal spectrum balancing for digital subscriber lines. In GLOBECOM - IEEE Global Telecommunications Conference,, November 2006.

[27] M. Cendrillon, R. Moonen. Iterative spectrum balancing for digital subscriber lines. IEEE International Conference on Communications, 2005. ICC 2005., 3:1937–1941, May 2005.

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

[29] G D Cioffi, J M; Forney. Generalized decision-feedback equalization for packet transmission with ISI and Gaussian noise, chapter 4, pages 67–79. Kluwer Academic Publishers, 1997.

[30] A.M. Turing. On Computable Numbers, with an Application to the Entscheidungsproblem. Proceedings of the London Mathematical Society, 1936.

[31] J Von Neumann. First Draft of a Report on the EDVAC. Reporting to United States Army Ordnance Department, June 1945.

[32] David L. McCubbrey Robert M. Lougheed. The cytocomputer: A practical pipelined image processor. In ISCA '80 Proceedings of the 7th annual symposium on Computer Architecture, 1980.

[33] Faye A. Briggs Kai Hwant. Computer Architecture and Parallel Processing. McGraw-Hill Book Company, 1984.

[34] Gordon E. Moore. Cramming more components onto integrated circuit. Electronics Magazine, 1965.

[35] R.K. Zhirnov, V.V.; Cavin. Limits to Binary Logic Switch Scaling–A Gedanken Model. Proceedings of the IEEE, November 2003.

[36] JM Rabaey. Scaling the Power Wall: Revisiting the Low-Power Design Rules. http://www.utdallas.edu/ vojin/classes/EE-7V82/Lectures/Rabey-PowerWall.pdf, November 2007.

[37] Various. Intel Core i7: Sandy Bridge overview. http://en.wikipedia.org/wiki/Sandy_Bridge, April 2011.

[38] M Flynn. Some Computer Organizations and Their Effectiveness. IEEE Transactions on Computers, September 1972.

[39] Y. N. Yeh, T.-Y.; Patt. Two-Level Adaptive Training Branch Prediction. In <u>Proceedings of the 24th annual international symposium on Microarchitecture</u>, 1991.