

FULLY PEER-TO-PEER VIRTUAL ENVIRONMENTS WITH 3D VORONOI DIAGRAMS

A thesis submitted in fulfilment of the requirements for
the degree of Doctor of Philosophy

MAHATHIR ALMASHOR

B.CompSc. (Hons)

School of Computer Science and Information Technology,

Science, Engineering, and Technology Portfolio,

RMIT University,

Melbourne, Victoria, Australia.

March, 2014

Declaration

I certify that except where due acknowledgement has been made, the work is that of the author alone; the work has not been submitted previously, in whole or in part, to qualify for any other academic award; the content of the thesis is the result of work which has been carried out since the official commencement date of the approved research program; and, any editorial work, paid or unpaid, carried out by a third party is acknowledged.

MAHATHIR ALMASHOR

School of Computer Science and Information Technology

RMIT University

19th March, 2014

Acknowledgement

My deepest gratitude goes towards the following people:

- My primary supervisor, Dr Ibrahim Khalil, who has stood by me and supported me throughout my candidature and whose patience is indeed at the level of saints.
- My secondary supervisor, Mr Geoff Leach, for the technical discussions into geometric algorithms, and more importantly, all the advice and support as I began my journey.
- Prof Zahir Tari, for his perseverance and continuous support throughout my candidature, and the School of Computer Science and IT, for the scholarships that sustained me during the early years.
- My beloved brother, Saifullizan Almashor, and dearest mother, Sarah Abdullah, without whom I would have literally and figuratively not even be here.

To all my friends and extended family, the gratitude within me is immense, and its receivers too numerous to mention succinctly. I love you all.

Credits

Portions of the material in this thesis have previously appeared in the following publications:

- M. Almashor, I. Khalil, Z. Tari, and A. Zomaya. "Automatic and Autonomous Load Management in Peer-to-Peer Virtual Environments," *Selected Areas in Communications, IEEE Journal on*, vol.31, no.9, pp.310-324, September 2013.
doi: 10.1109/JSAC.2013.SUP.0513028. (ERA A+)
- M. Almashor and I. Khalil. "Fully Peer-to-Peer Virtual Environments with 3D Voronoi Diagrams. *Computing*, Springer Vienna, vol.94, no.8-10, pp.679-700, Sept. 2012. ISSN 0010-485X.
doi: 10.1007/s00607-012-0197-9. (ERA A)
- M. Almashor and I. Khalil. "Timely Arbitrator Selection in P2P Virtual Environments with 3D Voronoi Diagrams," *Network Computing and Applications (NCA), 2011 10th IEEE International Symposium on*, pp.235,240, 25-27 Aug. 2011.
doi: 10.1109/NCA.2011.40. (ERA A)
- M. Almashor and I. Khalil. "Load-Balancing Properties of 3D Voronoi Diagrams in Peer-to-Peer Virtual Environments," *Parallel and Distributed Systems (ICPADS), 2010 IEEE 16th International Conference on*, pp.839,844, 8-10 Dec. 2010.
doi: 10.1109/ICPADS.2010.97. (ERA A)

- M. Almashor and I. Khalil. "Reducing Network Load in Large-scale, Peer-to-Peer Virtual Environments with 3D Voronoi Diagrams". *High Performance Computing (HiPC), 2010 International Conference on*, pp.1,10, Dec. 2010.
doi: 10.1109/HIPC.2010.5713197. (ERA A)
- M. Almashor, I. Khalil, and G. Leach. "Dynamic Game-Play Arbitrators with 3D Voronoi Diagrams," *Network Computing and Applications (NCA), 2010 9th IEEE International Symposium on*, pp.253,256, 15-17 July 2010.
doi: 10.1109/NCA.2010.46. (ERA A)

Abstract

The very notion of a fully Peer-to-Peer (P2P) Virtual Environment (VE) represents a unique challenge within the realm of networking and distributed systems. Online games are a prime example of VEs, forming the largest and most significant subset application. Whilst primarily for entertainment, their immersive virtual worlds have also become important social connection points, with successful titles boasting of player populations in the hundreds of thousands at any given time.

The scale of such Internet applications places exacting demands on the underlying network. The nature of online game-play places a premium on accuracy and speed in all player interactions, making games notorious for their sensitivity to network latencies and high bandwidth requirements. By aiming for a P2P architecture, and thereby omitting the centralised nature inherent in most commercial titles, the task of managing a disparate and dynamic network of peers is made ever more daunting.

Thus, resource scalability and system resiliency, hallmark characteristics of the P2P principle, are complicated with the need for secure and responsive game-play. It is not simply a matter of switching from a client-server (CS) implementation to that of a P2P one. Inter-peer communications need to be in almost real-time, to facilitate smooth game-play and virtual-world consistency. There is also a need to validate interactions, to ensure fairness and minimise cheating tendencies. Simply, players will not tolerate slow connections, nor cheaters that modify or delay inter-peer updates. The commercial success of any VE

depends on these key factors.

For these reasons, CS architectures have an established and overwhelming presence within VE space. It is far easier to simply restrict player connections to a centralised server, and thus control and validate the game-play from that single location. Issues pertaining to player/peer management, and the adjudication of player interactions (an important cornerstone of online games), are almost trivialised.

However, scalability is an issue within CS systems. As participant numbers soar, so does the demands placed on centralised services. Whilst we often hear of large player numbers for a particular game, the less obvious fact is that everyone is playing *separately*, in distinct slices of the virtual world severed from the rest. So, a player in Server-A may not necessarily interact with another in Server-B.

And thus, the *raison d'être* for a P2P-VE, and our efforts to eventuate it, becomes clearer. We seek to allow *all* peers to seamlessly connect with one another within a single, continuous and evolving virtual world. The crippling scalability limitations of CS systems are cast aside with a P2P overlay designed to efficiently connect a heterogeneous and volatile peer population to a vast and completely connected virtual world.

There are, of course, challenges to overcome. For any VE, arbitrating the interactions between players is an inescapable need. On-line games are equal parts collaboration and competition, requiring robust conflict resolution mechanisms to govern events between players. With CS, arbitration duties are simply assigned to pre-provisioned servers whereas in a P2P system, the issue looms large. As such, the facilitation of arbitration services across the peer population is the focus of the work contained herein.

We approach the problem by exploiting 3D Voronoi Diagrams (3D-VD) as a scalable, flexible and fault-tolerant P2P network overlay, aiming to fluidly distribute and manage arbitration services amongst peers. Traditional 2D varieties were often used as efficient area partitioning tools, and the novelty here lies in our utilization of a 3rd dimension (i.e.,

a Z-axis that accompanies the X and Y axes) which is able to embody non-spatial metrics. That is, rather than taking the Z-axis to literally mean above or below someone within the virtual map, it is used to signify other peer measures, such as current resource capacity.

The proposed approach effectuates a more fluid self-organisation amongst the potentially millions of peers within the VE. Work-loads of resource intensive arbitration tasks are handled dynamically at localised clusters. Speed of service is ensured as peers are able to easily appoint arbitrators from amongst fellow peers. The result is a decentralised mechanism that addresses responsiveness and security concerns, and mitigates the with the limitations of CS architectures.

Simulation results verify the feasibility and performance of our technique. It is shown how 3D-VD, with the right selection policies, can reduce arbitration failures (i.e., when a peer tries to obtain an arbitrator but fails). It is also shown that 3D-VD can appropriately distribute loads and reduce fluctuations by up to 90%. We also augmented the base technique with Newtonian gravity laws, providing an improved method to detect and respond to high-demand areas within the VE, and resulting in further reduction of failed arbitration attempts by 50%.

Contents

Declaration	ii
Acknowledgement	iii
Credits	iv
Abstract	vi
Contents	ix
List of Figures	xiii
List of Tables	1
1 Introduction	2
1.1 Online Games and Virtual Environments	5
1.2 Game-play Arbitrators	7
1.3 Scope and Limitations	9
1.4 Research Questions	13
1.5 Contributions	15
1.6 Thesis Structure	18
2 Preliminaries	21

2.1	The general P2P approach	22
2.2	Peer-to-Peer Virtual Environments	24
2.2.1	P2P in Games	27
2.3	First-Person Shooters	28
2.4	Dynamic Game-play Arbitrators	30
2.5	Voronoi Diagrams in P2P VEs	33
2.5.1	Clustering Peers	34
2.6	Dynamic Arbitrators using the 3rd dimension	37
2.6.1	Feasibility and Construction	38
2.6.2	Computational and Connection Complexity	41
2.7	Experimentation Details	42
3	Timely Arbitrator Selection with 3D-VD	46
3.1	Working with 3D-VD	46
3.1.1	Neighbour Clustering	47
3.2	Metrics for the Z -axis	50
3.2.1	Resource Capacity	51
3.2.2	Peer Reputations	53
3.2.3	Scale of Z -axis	53
3.3	Selection Policies	54
3.3.1	Timely Selection	54
3.3.2	Candidate Policies	56
3.4	Experimentation	58
3.4.1	Limitations	58
3.4.2	Simulation Set-up	60
3.4.3	Results	61

4	Network Traffic Reduction	71
4.1	Impact of The Z-axis	72
4.1.1	Metrics	72
4.1.2	Peer vs Global Maxima	73
4.1.3	Network Topologies	76
4.2	The issue of increased bandwidth	78
4.2.1	Neighbour Classification	79
4.2.2	Varying Upload Rates	79
4.3	Experimental Results	82
4.3.1	Test Set-Up	82
4.3.2	General Trends	83
4.3.3	Per Peer Reduction	87
5	Load Management	92
5.1	Background	92
5.1.1	Voronoi Diagrams	93
5.1.2	Role of arbitrators	95
5.2	Related Work	96
5.2.1	Player Flocking Mechanisms	98
5.3	Automatic Load Management	99
5.3.1	Using 3D-VD	99
5.3.2	Defining Load	101
5.3.3	Using the Z-axis	104
5.4	Experimental Results	105
5.4.1	Average Free Channels (AFC)	106
5.4.2	Fluctuations in Free Channels	109
5.4.3	Number of Neighbours	114

5.5	Flocking and Hotspots	119
5.5.1	Dissemination with 3D-VD	121
5.5.2	Simulation Parameters	122
5.6	Using Newton's Laws	125
5.6.1	Further Experimental Results	127
6	Conclusion	131
6.1	Research Aims	131
6.2	Contributions	133
6.3	Future Work	136
6.4	Final Words	138
	Bibliography	139

List of Figures

1.1	The different states of localised game-play, and how arbitration is needed for accurate and fair game-play in subsequent stages.	10
1.2	A top-down perspective of a virtual environment, illustrating the localised interactions between peers and their relation to events in the wider world. . .	11
2.1	Typical connection and game-play characteristics of online games.	26
2.2	Illustrates the inherent conflict of interest when deciding game outcomes and the role of arbitrators in securing game-play fidelity	30
2.3	An illustrative 2D Voronoi Diagram, with its dual structure the Delaunay Triangulation.	35
2.4	Extending from 2D Voronoi Diagrams: (a) reveals the 3D aspect by exposing the Z-axis; (c) demonstrates the resultant partitioning when the actual 3D Voronoi Diagram is derived from the peer positions.	38
2.5	3D-VD derivation times	40
2.6	Breakdown of 3D-VD computation	41
3.1	Illustrates the differing in-game view of peers within the P2P-VE when using 2D (top), as opposed to 3D Voronoi Diagrams (bottom). It is seen how the added exposure from the Z-axis allows the computing peer to see and link with previously hidden neighbours.	47

3.2	Showing the 3D version of the Voronoi pruning process adapted from [Hu et al., 2006]. <i>Note: The actual polyhedral Voronoi cells were removed to enable visualisation of the peers.</i>	49
3.3	Illustration of how a peer derives its own 3D-VD representation of the others around it, decides on its neighbours (NB) and subsequently ranks them in terms of suitability. Nodes seen here were generated for illustrative purposes.	55
3.4	Performance of 2D Euclidean-distance policies. From top-left, clockwise: Near-est , Central and Outmost . Their 3D counterparts are not shown as plots very closely follow the results here.	61
3.5	Failure rates for Mass-effect selection policies (static on left and dynamic on right) against C2D.	62
3.6	Performance of “Fittest” and the baseline random (RAN) policies.	63
3.7	A comparison of <i>Tries</i> (on left of each <i>column pair</i> , red stripes) against successful arbitration (on right, solid green). Top row shows the server-less peer configurations, whilst the bottom row shows the general performance of server-enabled configurations.	64
4.1	A 3D View of a mixed, server-enhanced peer topology consisting of 10% servers (orange cubes at top), 20% Super-Nodes (blue cones), 50% Normal Nodes (green dots) and 20% Weak Nodes (crosses) is shown in (a) . (b) further exposes the Z-axis to better illustrate the distribution of peer types according to current bandwidth capacity.	75
4.2	Shows a mixed-egalitarian topology where only normal-nodes and weak-nodes exist in a 50-50% distribution. Note the absence of any high-capacity peers inhabiting the top of the cube.	76

4.3	Showing the dual-tiered classification of the neighbours (NB) of a peer and their relative numbers. The peer computing its 3D-VD is the central red sphere. The blue cubes in (a) represent the <i>enclosing</i> NBs whereas the yellow cones in (b) are the <i>boundary</i> NBs. (c) shows the fully interconnected graph (note: nodes are illustrative).	81
4.4	Probability Mass Functions of recorded upload bandwidths for completely server-less topologies	84
4.5	PMFs of configurations with minimal servers present (1 and 4)	84
4.6	PMFs of configurations with 40% Supernodes (Super-node Enhanced) and 10% Servers (Server-Enhanced)	85
4.7	Probability mass functions of three representative topologies running the enhanced updating algorithm. Note the general mean BW values as opposed to the ones just earlier (0.3 to 0.35 compared with 1.5 Mbps)	85
4.8	Cumulative Distribution Functions of three representative topologies, where the left most plots display the reduced bandwidth requirements and the plots on the right are their original recorded values.	86
4.9	Illustrates the bandwidth (BW) reduction per Peer and per Tick. Here, we see boundary neighbours (NB) are sent updates at the same rate as enclosing NBs.	88
4.10	Illustrates the subsequent reductions shown with our approach, with the average BW per peer (solid horizontal line) dropping to well below the theoretical limit (dotted line).	88
4.11	The peers shown here are taken from a different configuration, where 10% of peer population are servers (accompanying reduction plot is omitted).	89
4.12	Reduction comparison across all topologies	90
5.1	Overview of various subset application areas in DVE research.	93

5.2	A 2D Voronoi Diagram (left) and the enhanced 3D version	94
5.3	Illustrating the inherent conflict of interest in typical FPS interactions and the intended role of arbitrators as dynamic referees.	95
5.4	Each peer derives its own 3D-VD view of the world around it, decides on its neighbours (NB) and subsequently ranks them in terms of arbitrator suitability. This happens at every tick of the simulation, for all peers. [Almashor and Khalil, 2010b]	100
5.5	The probability mass functions of the Average Free Channels (AFC) per peer, which are analogous to the average positions of peers along the Z -axis. The plots are for a configuration of 1000 peers with egalitarian make-ups (i.e., no servers or super-nodes). (a) shows performance with a randomised arbitration policies, whilst (b) shows the fittest arbitrator policy in action.	107
5.6	Similar to Fig 5.5, the plots here show the PMFs of per peer AFCs. These have configurations of 2000 peers that are enhanced by super-nodes . (a) shows performance with a randomised arbitration policies, whilst (b) shows the fittest arbitrator policy in action.	108
5.7	PMF plots of per peer AFCs for configurations of 2000 peers with enhancement by servers this time. (a) shows performance with a randomised arbitration policies, whilst (b) shows the fittest arbitrator policy in action.	109
5.8	Cumulative distribution functions (CDF) of the average free channels fluctuations (AFLUX) experienced by peers. Shown here are configurations with 1000 peers containing 10% super-nodes.	110
5.9	CDFs of the peer AFLUX values. Shown here are configurations with 2000 peers containing exactly 4 high-capacity dedicated servers.	111

5.10	Comparison of server workloads between Random Arbitrator Selection (RAS, top) and Fittest Arbitrator Selection (FAS, bottom) policies. Configuration with 4 servers are shown as none of the other peer configurations contained servers.	112
5.11	Shows the proportion of arbitration workloads that servers process in relation to the rest of the peer population. The FAS policy is shown on the left, whilst RAS is shown on the right. Again, only server-enabled peer configurations are included here.	113
5.12	The Mean Number of Neighbours (M-NNB) across all configurations are displayed here. On the left of each plot shows results when FAS has been activated, whilst RAS is shown on the right.	115
5.13	Illustrating the effects of peer layering and exposed locations in a 3D-VD. Server-1 and Server-2 are the square and dot, respectively. Normal-nodes are triangles with diamonds being weak-nodes.	117
5.14	Differences seen when routing help-request packets with 2D-VD (in red) and with 3D-VD (in green).	121
5.15	Showing the order, placement, dispersal and type of simulated hotspots used in our experimentation.	123
5.16	Illustrating how servers are pulled towards hotspots.	124
5.17	Breakdown of force vectors	125
5.18	Showing decrease in failure rates when activating the extended ALM technique alongside 3D-VD.	128
5.19	Hotspot dynamics with ALM (top) and without (bottom)	129

List of Tables

2.1	Comparison of the most popular game-types (with commercially availability)	29
3.1	Details of simulated peer populations and their notations in later results. . .	59
3.2	Explanation of Event Types (as shown on the X-axis of all results graphs)	59
3.3	Summary of related architectures and approaches	68
4.1	Comparison of simulation topologies	74
4.2	Expected Number of Neighbours	78
5.1	Peer Configuration Codes and Details	103
5.2	Showing typical data within our Help-Request Packets	127

Chapter 1

Introduction

Humanity has an almost inescapable need to be fully immersed in their environments. With a world increasingly lived online, Virtual Environments (VE) are our first tentative steps towards fulfilling that need. The idea is of a persistent virtual world which everyone connects to and participates in. Distributed or Networked Virtual Environments (DVE/NVE) [Hu et al., 2006, Claypool and Claypool, 2006, Steed and Oliveira, 2010] are the official academic terms, encompassing varied concepts such as virtual conferencing and tele-presence. However, to the general public, this field of study is exemplified by its most famous application: **online games**.

The overarching goal here is to work towards a fully Peer-to-Peer Virtual Environment (P2P-VE). That is, the implementation of an interactive virtual world¹ on strictly decentralised architectures. Continuing public fascination with such networked applications and particular interest in online games for the younger demographic, precipitate the need from within both industry and academia to address two intertwined game-play issues: security and responsiveness.

The underlying concept linking these two inseparable requisites is that of *game-play*

¹often characterised by massively multi-player online games (MMOG) such as World of Warcraft, Everquest and Guild Wars

arbitrators. These are the adjudicators within the virtual world, transient referees that decide on the outcomes of interactions (e.g., combat or in-game commercial transactions) amongst peers. They fulfil the inherent need for security, ensuring game-play *integrity* between competing peers, and also enhance responsiveness by employing latency mitigation techniques when resolving said interactions.

Online games [Ross, 2009, Achterbosch et al., 2007], have their roots in Distributed Virtual Environments (DVE) [Kwok and Wong, 2008], itself an extension of the earlier Networked VE (NVE) [Macedonia and Zyda, 1997]. Derivatives are present in myriad of other fields, such as the military [Macedonia, 2002, Zyda et al., 2003], social networking [Macedonia, 2007, Messinger et al., 2009] and collaborative tools [Boukerche et al., 2010, Rhee et al., 2007]. Our concept of a fully P2P-VE encompasses both classifications and is driven in part by prior work in [Hu et al., 2006, Rueda et al., 2008]. It strongly defines an underlying network architecture that is fully decentralised with regards to bandwidth, processing, management and storage requirements.

Indeed, more popular games such as World of Warcraft and other Massively Multiplayer Online Games (MMOG) exemplify an extreme from of DVEs. They combine the challenge of reliably connecting large numbers of concurrent players (typically 100,000 and above) with the need to ensure *secure* and *responsive* game-play. In general, players within a VE require:

- assurances that the integrity of the game is not compromised by cheaters
- timely mechanisms allowing smooth, accurate and immersive game-play

Game developers and publishers have focused more on centralised mechanisms [Kushner, 2005, Claypool and Claypool, 2006, Ross, 2009], preferring that their customers connect to centrally managed infrastructure for access to their games. This is in line with industry needs to control, manage and monetize their products. Our approach here treads an

alternate path, striving towards a complete decentralisation of the underlying network to create a fully Peer-to-peer Virtual Environment (P2P-VE).

It should be noted briefly that the term fully P2P-VE may be considered at the extreme end of the spectrum with regards to P2P systems. The aim here is to strive for full decentralisation, and not for a solution placed halfway between P2P and centralised systems (i.e., hybrid or limited P2P networks). This definition will be further explored in later chapters, most notably in 2.

Consequently, the task at hand becomes considerably more difficult. How do we connect such large numbers of players in a scalable manner? How do we address the aforementioned security and game-play issues? How do we manage and control an inherently distributed system with little to none in the way of centralised tools?

The work herein strives to answer these questions, with the outline of our contributions being as follows:

- A proposal to utilize 3D Voronoi Diagrams (3D-VD), a fundamental geometric construct, to dynamically cluster and manage peers in a completely decentralised architecture as unique as those typified by P2P-VEs.
- An explanation of our novel augmentation to basic 3D-VDs, where the typical spatial 3rd dimension often used is replaced instead with pertinent network metrics that enhances the performance of our network overlay.
- A technique that uses 3D-VD as a way to appoint dynamic game-play arbitrators on the fly, and in a timely fashion, thereby aiding in maintaining VE consistency and responsiveness.
- Further augmentation to the base 3D-VD method, in order to minimize exponential bandwidth utilization growth amongst the peer population of P2P-VE.

- The discovery and analysis of natural load-balancing properties of the base 3D-VD technique, and the subsequent improvements made to enhance load management performance in a decentralised system.
- Extensive analysis and discussion of obtained simulation results and feasibility studies, and comparisons with prior works in the field of DVEs.

The above issues and our contributions become more pronounced when we consider the sub-genre of on-line games that we target: that of the First-Person Shooter (FPS). This popular form of online game demands quick reflexes and timing on the part of the player, with speed and accuracy important prerequisites to winning the game. This form of game-play places a premium on timely inter-peer communications [Claypool and Claypool, 2006], and further complicates earlier proposals and established industry methods.

The reader is advised that throughout the work contained herein, the terms “node, peer, player, participant and avatar” are used interchangeably. They refer to a distinct entity within the network, and generally is a host computer with which a human player participates in the virtual world. An arbitrator is a higher-level abstraction of the required functions of certain nodes within the network, and can be both permanent (e.g., a dedicated server always performing arbitration) or dynamic (a peer performing similar duties).

1.1 Online Games and Virtual Environments

Multiplayer Online Games (MOG) such as Everquest [Kushner, 2005] form the largest subset of Distributed Virtual Environments, and are still built on traditional centralised principles. There are yet to be realised within a fully Peer-to-Peer (P2P) network architecture. While the P2P paradigm offers many desirable features such as processing scalability and excellent fault-tolerance [Skibinsky, 2005, Knutsson et al., 2004], it also introduces challenges that have yet to be overcome by the academic community.

Chief amongst these concerns is the demanding network requirements of a P2P-VE. A naively built system results in quadratic growth of bandwidth usage, with each and every peer expected to communicate *with every other peer* [Hu et al., 2006, Bharambe et al., 2008]. Adding to this issue is the increased sensitivity to high latencies that certain classes of VEs exhibit [McCoy et al., 2005, Safaei et al., 2005]. Games such as Everquest are more tolerant of latency issues [Chen et al., 2009], whereas First Person Shooter (FPS) games exhibit some of the tightest upper-bounds [Zhou et al., 2008]. This is due to the mechanics of FPS games, where a player must aim quickly and accurately at opponents in order to succeed. Thus, in such games, timely updates between participants is of the utmost importance.

These factors contribute greatly to one of the primary issues with P2P-VE: the crippling network load at each peer and, by extension, the whole system. The volume of traffic generated quickly outstrips the capabilities of typical peers in the network. As reported in [Bharambe et al., 2008], values as high as 10 Mbps are reported, which far exceed the *upload* capacities of modern broadband home connections. Addressing this issue is the primary focus of our work here.

The classical approach is to apply interest management mechanisms such as those proposed in [Keller and Simon, 2002, Steed and Angus, 2005, Bharambe et al., 2008]. These are concerned with limiting the communication scope based upon past interactions, the current attention of the human player and in-game visibility ranges. The issue is thus generalised to that of *clustering* the participating peers within the P2P-VE. Doing so allows for a particular peer to communicate with only a smaller subset of his fellow peers. High-level hybrid architectures, a varying amalgamation of P2P and Client-Server (CS) paradigms, have also been regularly proposed [Skibinsky, 2005, Rooney et al., 2004]. However, in these hybrid systems, there still exists a need for dedicated servers to manage the peers.

The *filtering by proximity* approach by Hu *et al* [Hu et al., 2006] where in-game (i.e.,

virtual-world) coordinates are used to group peers is most promising. Their use of basic two-dimensional Voronoi Diagrams (2D-VD) addressed the network traffic scalability problem. Our prior work in [Almashor et al., 2010] was in an effort to improve and overcome drawbacks introduced when using 2D-VD in P2P-VEs via the introduction of a *third dimension* in the VD computations (3D-VD).

3D-VD is a promising approach when considering the other pressing issues surrounding P2P-VEs, such as (i) game-play responsiveness; (ii) fair and secure game-play resolution and (iii) load balancing. It effectively allows an additional avenue with which to tackle these issues and thus, is a much desired feature. In our wider experimentation, we note several system-enhancing properties of 3D-VD. One such unique outcome is the focus of this paper; that is, when combined with a peer classification mechanism, 3D-VD exhibits the ability to significantly reduce bandwidth usage in fully P2P-VEs.

1.2 Game-play Arbitrators

Thus, a P2P-VE may be considered to represent a supreme category of classical DVEs. There are of course, other game-play forms that do not require the stringent requirements of FPS games, and thus places more relaxed preconditions onto any decentralised on-line game. However, the focus here is to solve issues at the extreme end of the game-play spectrum, in the hopes that any solution there would be easily transferable to the less demanding game sub-genres.

As stipulated before, the issue of game-play arbitration is of paramount importance. The manner in which we resolve interactions between peers (combat or otherwise) is made ever more significant in an architecture with no centralised control mechanisms, and expects an efficient solution to the issue of inter-peer connections. In order to proceed, the work here addresses the core issue of the *timely selection* of these needed arbitrators. They are the vital cog in the machinations of an on-line game, and are central to any effort to

secure game-play and they improve VE responsiveness. Thus, initial thoughts lead to the following questions:

- How do we select suitable candidate referees from amongst the total peer population?
- More crucially, how do we perform this selection without negatively impacting responsiveness?

Moreover, this arbitrator selection process is to be performed in a completely decentralised manner, in keeping with the concept of a fully P2P-VE. Furthermore, it has to curtail any latencies introduced during the process and minimize any impediments to the flow of a game. In effect, we are striving to *choose the right candidate first* as we can ill-afford multiple attempts to re-negotiate the appointment of alternative peers as arbitrators.

There is simply no room for a more relaxed process, especially in our targeted sub-domain of FPS games. When a player is engaged in a heated battle against another player, the arbitrator needs to be appointed *now*; it needs to communicate with the combatants *now*; it needs to perform its arbitration duties *now*. Any mechanism that delays the arbitration process cannot be effective within a P2P-VE. For example, a system that polls multiple peers, selects a candidate, tries to connect, fails and re-tries would introduce latencies and increase bandwidth usage amongst peers and ultimately, be doomed to failure.

Group voting and result validation techniques such as those proposed in [Baughman et al., 2007, Corman et al., 2006, Webb et al., 2008, Goodman and Verbrugge, 2008] incur greater overheads and are, in our humble opinion, best avoided. Additionally, there exists the paradox with such schemes whereby the game-play decision or interaction result may ultimately rest with a single peer. For example, if a group of 5 judges were to vote 2 against and 2 in favour of a result, would not the deciding vote be the purview of a single peer? And if so, would it not be better to appoint a *single* deciding peers instead of a group?

Thus, the goal is further refined to the selection of a singular arbitrator in the shortest amount of time. Consequently, there is a great emphasis on the criteria used in the selection process. Several questions arise, namely:

- Should we choose the nearest peers? Or the furthest?
- What about those with more available resources?

Accordingly, we develop several selection policies, from simple euclidean distance measures to novel ones based on Newtonian mass effect laws, taking advantage of our implied 3D domain. We investigate the implications of each of our designs and analyse our obtained simulation results. This, along with the use of 3D-VD and non-positional metrics, is one of the main contributions of the work contained herein.

1.3 Scope and Limitations

The topical coverage when attempting to enable a fully P2P-VE can be quite large, as we are effectively marrying 3 distinct research areas, namely: (i) on-line virtual environments, (ii) network overlays, and (iii) P2P architectures. As prior sections have alluded to, we are primarily concerned with the question of arbitration services, and more specifically, their discovery and management amongst a heterogeneous peer population.

There remains however, the question of the target application and the specifics of operation for an on-line game. Whilst we do target a very specific subset of virtual environment (i.e., FPS games), there is a need to explain our intended angle of approach, and the accompanying scope and limitations.

The concerns addressed with the 3D-VD technique relates only to *localised game-play* within the virtual world. In a typical on-line combat game, there will be frantic yet short-lived interactions amongst players. For example, a participant may travel the virtual map

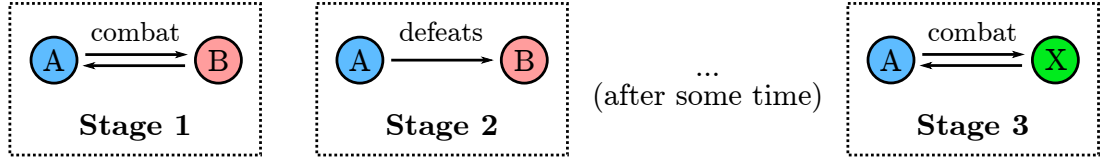
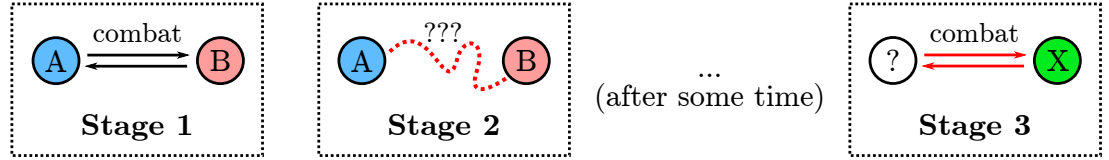
With correct arbitration**No arbitration or compromised**

Figure 1.1: The different states of localised game-play, and how arbitration is needed for accurate and fair game-play in subsequent stages.

for a relatively long period of time, before encountering enemies and engaging in frenetic combat for a short period of time (usually in the magnitude of mere seconds). Of course, other events may be happening elsewhere in the game. However, to that player alone, nothing happens as he travels towards the combat area. This is then followed by a burst of activity as he battles opponents and the game is resolved one way or another.

These localised interactions are key. We can easily distil the mechanics of an entire virtual world and eventually, come to the realisation that such interactions are the atomic pieces of events that drive game-play. Cumulatively, these separate instances of game-play are what actually moves the whole game forward in time (within the constraints of the VE mechanics), and that the consistency and the continuity of the virtual world depends on the accurate, fair and timely computation of such localised game-play.

This is illustrated in Figure 1.1, where we see an inaccurate or corrupted game-play resolution affecting later events within the virtual world. With timely and fair arbitration services available at stage 1, Players A and B engage in combat. Subsequently in stage 2, Player A emerges as the victor from the interaction, and continues participation within

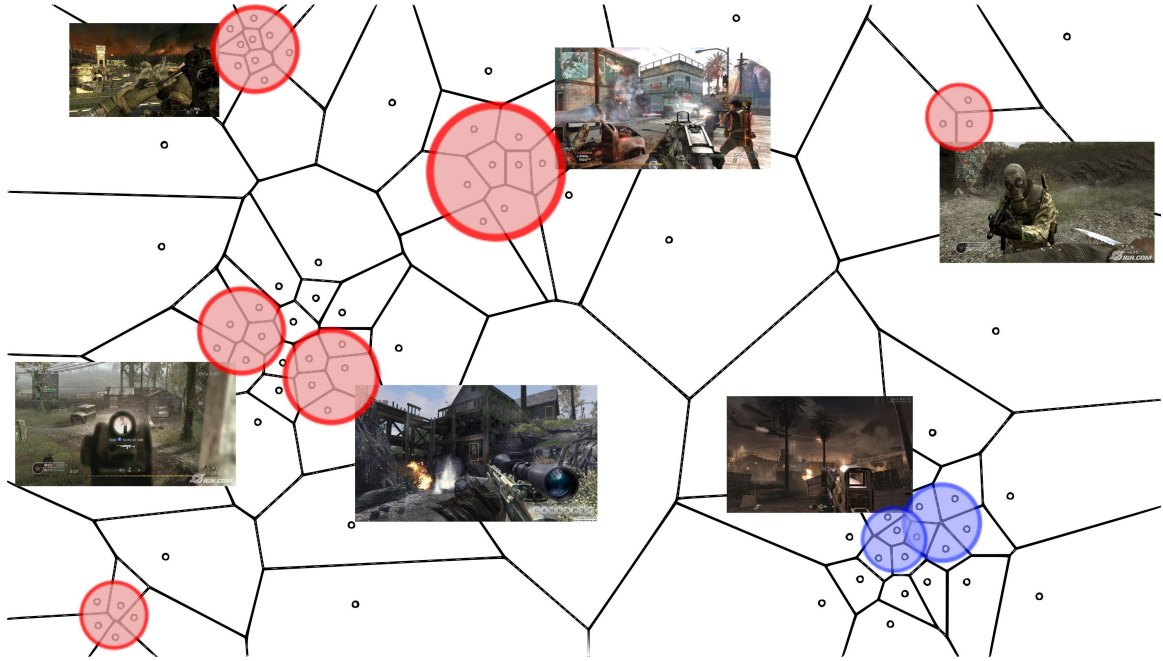


Figure 1.2: A top-down perspective of a virtual environment, illustrating the localised interactions between peers and their relation to events in the wider world.

the game, whereas Player B, being defeated is removed (perhaps to be respawn elsewhere in the game-world, according to the mechanics at play). After a length of time, Player A encounters another opponent in X, and begins hostilities with it.

When there is little or no arbitration at these almost atomic, localised levels, it is seen that game-play further along in game-time will be affected. As we can observe from the lower half of Figure 1.1, issues with arbitration services at stage 1 inevitably results in discrepancies and paradox situations which potentially undermines the entire game.

Figure 1.2 show examples of these localised combat events, and how it affects peers within close proximity to each other. Peers in other areas of the map are absorbed within their own conflicts, and do not necessarily care about the machinations of other players. These are the dynamic, *tick-by-tick* events that, while cumulatively having a large impact on VE consistency, are actually small pieces of interactions.

Accordingly, the data and the bandwidth required at such a small scale is predictably limited. Data exchanged between peers and the arbitrating server during localised game-play can be measured within the order of kilobytes, with individual packets usually within 100 bytes in size. The following code sample shows the typical data sent from a peer to its connected game server to indicate its move for a simulation time-step.

```
struct PlayerMove
(
    float TimeStamp,
    float AccelX,
    float AccelY,
    float AccelZ,
    float LocX,
    float LocY,
    float LocZ,
    byte MoveFlags,
    rotator Rot,
    int ViewPitch,
    int ViewYaw
);
```

Typical game-software that is installable on the players' home computers can range from the order of hundreds of megabytes and beyond. Indeed, with art assets (e.g., graphics textures and models) and sound clips, current commercial games have been known to exceed 10 GB in size. With developers chasing increasing complexity and richer graphical details within their games in order to attract more users, software sizes can only increase further.

The scope of our work here however, is not concerned with the the delivery, validation and management of such data. The work here does not address issues pertaining to high-bandwidth usage and/or the storage amongst peers. This in itself is somewhat a significant diversion from traditional P2P research, which the reader may reasonably assume from the title of the work and the frequent repetition of the term P2P-VE.

The assumption here is that all peers are already fully-equipped, with the large game assets pre-installed and obtained through external means. We do not consider the download

and delivery of such game data, and are more attentive to the state-change data detailing the interactions amongst peers (which are small but frequent in nature). In all our experimentation, we assume all peers have up to date software. Whilst we are interested in the validity of game-client with regards to potential cheating, the issue is defined as within the constraint of selecting and utilizing an effective arbitrator (e.g., a server or fellow peer).

1.4 Research Questions

The use of 3D-VD in the area of VEs and in general network research is novel and not without its challenges. We are concerned with extending basic 2D-VD usage with the third dimension, in an effort to better cluster the peers. In addition, we aim to supplement or possibly replace the use of in-game coordinates with other non-positional metrics. Most current research has concentrated on the use of strict positional, in-game coordinates to derive their VD representations of the game-world. We see the potential in the use of 3D-VD to achieve better results and address shortcomings with the prior approaches within the sphere of P2P gaming.

As with any new tool, 3D-VD needs careful study and analysis before its application in the targeted domain. While there is undoubtedly great potential with Voronoi Diagrams, the approach here is more conservative and seeks to first understand its initial use in P2P-VE. Indeed, we could take the concept further by for example, extending the core 3D-VD construct with multiple non-spatial axes, or using cumulative and composite metrics, and even extending to a 4th dimension. However, the task at hand is to first explore, understand and apply 3D-VD in limited settings, and pave the way it to be used as a potential viable P2P network overlay. Thus, the aim of the work contained herein is to address the issues arising from the following research questions:

1. Computational Performance

What are the implications of using Voronoi Diagrams with higher dimensions? The very act of deriving a 3D-VD can be considered a detriment to the approach. As with any computational geometry problem, the mere addition of an extra dimension bodes ill for the timeliness of the process. So, can 3D-VDs be derived efficiently and in a timely manner? How will the extra dimension impact our derived VDs?

2. Using 3D-VD Effectively

This subsequent step is perhaps the most important foundational investigation into 3D-VD. Assuming an ability to efficiently compute 3D-VD, how do we actually employ it within a P2P-VE framework? A primary concern would be suitable metrics for the new Z-axis and how best to utilize the flexibility and clustering performance of 3D-VD. And once successful, what schemes and policies can we use to select arbitrators?

3. Maintaining Scalability

An obvious by-product of opening up a third dimension would be more exposure to adjacent peers in close proximity of the central peer. This means more network connections, potentially creating a larger bandwidth burden on all participants. How can we counter such increases in bandwidth requirements? Is it possible to employ 3D-VD, yet still maintain a fluid P2P-VE?

4. Improving Load Management

Effective use of 3D-VD, especially coupled with suitable arbitrator selection schemes, results in natural load-balancing properties. Load management is a perennial and paramount issue in distributed environments, more so in general P2P research. Here, we examine 3D-VD's aptitude in load-management and look to overcome the player flocking phenomena and ascertain the most effective arbitrator selection policies.

1.5 Contributions

The four research aims outlined above forms the core of the work prepared within. There are, as alluded to earlier, more interesting and challenging issues in the application of 3D-VD for P2P-VE use. However, the work here advances the art by forming an imperative of the basic, initial steps in its implementation. The contributions are as follows:

1. Analysis and validation of 3D-VD

The computational cost of deriving 3D-VDs is an important consideration, especially when we take into account the latency-sensitivity of on-line games and general VEs. Game-play responsiveness is vital to the immersion of players within the virtual world. Any protocol that negatively impacts the speed of the application to provide feedback to players needs to be avoided.

The most urgent question would be if a 3D-VD can be derived efficiently. 2D representations may be computed in a timely manner, up to the order of thousands of peers. However, as is the case with most computational geometry problems, deriving higher order VDs becomes progressively harder and computationally intensive with each additional dimension. Pursuant to this, we also need to investigate the impact of using these extra dimensions and monitor any negative implications. For example, a poorly defined metric chosen for the 3rd dimension may cause peers to artificially clump together and thus, degrade the usability of the VD.

Early experimentation proved promising, with 3D-VD derivation times coming within the needs and constraints of FPS games on current computing hardware. An extensive analysis was performed, with much work dedicated to the implementation of a 3D-VD simulation engine created in C++ that then served as an experimental platform for out ensuing investigation.

2. Effective arbitrator selection policies, and metrics for the 3rd dimension

The question remains as to what metrics can be utilised as substitutes to in-game coordinates (spatial X-Y positions). Using virtual world locations provides a natural clustering method based on the spatial relationships amongst peers. However, a fundamental drawback exists in that peers are forced to maintain direct connections with only its closest neighbours. This is inadequate when we consider that there may be two peers who are separated by a large distance but still retain an interest in each other. 3D-VD addresses this issue yet the issue is what to nominate as the metric for the Z-axis?

It cannot simply be assigned as the height of a peer within the game (e.g., the altitude of a player in relation to others within the virtual world). Doing so defeats the utility of 3D-VD. Thus, different quantitative measures were explored and peer load and capacities was identified to be a good substitute into this initial foray into fully P2P-VEs. The choice was made to better manage the use of resources and hardware limitations at each peer, and to pursue the P2P objective of better load management. Experimentation into appropriate arbitrator selection schemes, and the subsequent results, also proved a crucial component to our approach. 3D-VD in itself can be surmised as a network overlay, and though an interesting and flexible one, it still exists purely to connect peers. It is what we do with the links it affords that matters. Accordingly, extensive work was put into comparing competing selection policies for arbitrators within the sphere of neighbours of the computing peer, with interesting outcomes emerging from our analysis.

3. Bandwidth reduction model when utilizing 3D-VD

With more exposure along the Z-axis, comes more inter-peer connections. The mere fact that peers can be arranged in a 3-dimensional domain instinctively means that each peer sees more of the world around it and by extension, more neighbouring

peers. With 2D-VD, this effect is constrained by surrounding neighbours occluding further-away peers on a 2D plane. Using 3D-VD however, introduces more linkages amongst peers, which in an already constrained environment such as an on-line game, bodes ill for the functioning of the P2P-VEs.

The issue here is with increased bandwidth costs to maintain said inter-peer connections. A peer sending frequent updates to 10 peers in a 2D-VD enabled platform could now find itself potentially sending those same frequent updates to *100* peers instead. With current limitations on home broadband access, this effect will likely saturate the outgoing capacity of a peer and is inherently detrimental to the functioning of the P2P-VE.

Thus, a mechanism was implemented in conjunction with 3D-VD to minimise the effect of the bandwidth upsurge. A two-tier model was developed that intelligently reduced updates to further away neighbours while maintaining a more frequent sending rate for neighbours with a closer proximity. This is so as to maintain VE consistency and cohesion, yet still reduce the bandwidth requirements when applying 3D-VD.

4. Automatic and autonomous load-balancing mechanism

With a flexible platform in 3D-VD, and an effective policy to impact the selection of suitable arbitrators, it was discovered that our approach had strong load-balancing properties that positively influenced the functioning of a P2P-VE. Experimental results indicated that load was effectively being shared amongst all peers within the virtual world. Simply put, arbitration load, which represents an increase above and beyond normal updates performed to maintain inter-peer connections, is always directed to the most suitable candidate.

This inherent load-management is a consequence of using 3D-VD and thus, having the visibility and flexibility afforded by the Z-axis. By always seeking peers with

maximum capacity as arbitrators, we are effectively directing load to where it is best able to be coped with. This effect is studied and analysed within the confines of our domain with special attention given to a variety of peer population make-ups. These include an egalitarian configuration with no high-capacity peers, to model a scenario where all peers within the P2P-VE are constrained. Another tested configuration had minimal amount of servers introduced into the system, to model a scenario where the developers wished to add to peer capabilities.

Yet, for all the inherent goodness of this automatic load-balancing, it is still a somewhat static mechanism. Player flocking is a phenomena not unlike flash-crowd problem, where websites experience a sudden surge in demand that soon abates. This causes issues with the proper functioning of the websites, and even failures. Likewise, the players in a VE may flock towards a certain area of the map, thereby increasing the load there. This concentration of activity cannot be handled by the base 3D-VD mechanism and thus, we developed an autonomous method to guide high-capacity peers (or dedicated servers) towards such problem areas. Doing so reactively alleviates the stress in that location and leads to better game-play experience for all peers within that vicinity. The experimentation performed produced good outcomes, and these were analysed at length to explore the utility of applying such an autonomous mechanism.

1.6 Thesis Structure

The subsequent text can be categorised as follows:

- **Chapter 2** provides a preliminary discussion on the concept, feasibility and use of 3D Voronoi Diagrams. The approach here is general and focuses on prior art relating to Voronoi Diagram use, on-line games and P2P-VEs, with specific aims to introduce the reader to 3D-VD and validate its utilization.

- **Chapter 3** deals with the actual application of 3D-VD, the metrics used for the Z-axis, and the arbitrator selection policies (e.g., fittest, random, proximity, mass-attraction, etc.) used in all our simulations. This allows the reader a firmer understanding on how 3D-VD functions and how it enables a fully P2P-VE.
- **Chapter 4** aims to address a paramount side-effect resultant from the application of 3D-VD, the surge in bandwidth requirements from the additional linking of peers. A two-tiered methodology is applied to control the upload demands whilst still maintaining a cohesive and responsive virtual world.
- **Chapter 5** explores the inherent load balancing features of 3D-VD, weighing the effects of various arbitrator selection schemes against each other and discussing the spread of load, and the minimisation of load fluctuations. The issue of player flocking is also addressed, with additional experimental results analysed.
- **Chapter 6** concludes the work contained herein, allowing us to summarise our contributions and discuss future avenues of research.

Readers should note that we've strived to be direct and relevant to the subject matter. The aim was to be more concise, and pursue of the question of 3D-VDs and their applicability to the area of on-line games and virtual environments. Extraneous avenues of research were omitted to focus solely on the exploration of 3D-VD. Thus, the concepts touched on, and the experimentation undertaken, worked towards this goal. Each simulation and subsequent analysis within chapters 3, 4 and 5 aids us in our exploration of 3D-VD as a platform for P2P-VEs.

As mentioned earlier, a key issue in the operation of virtual worlds (especially competitive ones) is that of arbitration. Accordingly, we tackled this head on in chapter 3, with experiments to ascertain the feasibility of deploying 3D-VD in such scenarios. The questions we sought to address related to both the suitability and the actual mechanics of

3D-VD. That is, how do we use the newly created Z-axis? What metrics can we successfully apply? More importantly, once applied, we tested the suitability of each proposed arbitration selection policy.

This led to the discovery of an unwanted phenomena: the increased demands on bandwidth on all peers within our simulation environment. In brief, applying 3D-VD meant that peers were communicating with one another far more frequently, in an effort to support a fully P2P network. This surge resulted from the need for peers to keep track of one another and maintain consistency within the virtual world. As such, we devised a method to reduce the bandwidth costs, whilst still maintaining direct inter-connections amongst all peers. The simulations performed in chapter 4 were focused solely on this issue and to test the feasibility of our two-tiered connection technique.

Finally, in chapter 5, we explored and tested algorithms that supported the natural load-management capabilities of 3D-VD, and experimented with ways to assuage issues rising from sudden surges in bandwidth demands. This follows from earlier experimentation, that is, once timely arbitrator selection and bandwidth costs were dealt with, how do we manage the arbitration work-load amongst peers? The simulations done in this chapter focused primarily on this issue and served to augment the results obtained in prior chapters.

In summary, the experimentation contained within each of these three chapters seek to address, in a linear fashion, the following higher-level issues:

1. How rapidly can we select arbitrators?
2. Can we reduce crippling inter-peer bandwidth?
3. How can we spread the load amongst arbitrators?

Chapter 2

Preliminaries

This chapter affords the reader a basic familiarity with the underlying concepts needed to achieve our stated aims of a fully P2P virtual environment. The overarching premise of a P2P-VE is deconstructed into specific research areas, and we examine the scopes and limitations within each to provide a better of understanding of the system as a whole.

Peer-to-Peer (P2P) networks for Massively Multi-player On-line Games (MMOGs) have been proposed in academia regularly ever since the latter's rise as an economically viable interactive medium. MMOGs, which currently form the largest subset of Networked or Distributed Virtual Environments (NVE/DVE), are widely tipped to be one of the main engines for future growth in on-line services [Hu et al., 2006, Kushner, 2005, Chen et al., 2009].

The academic impetus to adopt P2P principles in MMOGs is driven, in part, by the need to counter several issues with the current client-server (CS) paradigm, namely: (i) bandwidth non-scalability, (ii) processing non-scalability and (iii) poor fault-tolerance [Skibinsky, 2005, Knutsson et al., 2004]. However, on the part of industry¹, the predominant architecture used when creating MMOGs and DVEs is still that of the traditional CS.

The primary issue with the application of P2P techniques to the area of MMOGs is essentially the peer-clustering problem [Hu et al., 2006, Morillo et al., 2007]. Simply, when

¹inclusive of game developers, game publishers and other VE operators (such as Linden Labs, creators of Second Life, a popular non-gaming VE)

given a population of peers within a Virtual Environment (VE), the question becomes: how do we maintain efficient, scalable communications amongst them, without the use of a central server authority? Each and every peer *cannot* maintain connections to *every other* peer.

Thus, the natural solution is to effectively cluster them into separate and distinct groups. This leads to the popular approach of a hybrid system incorporating CS and P2P elements, as in [Rooney et al., 2004, Liu and Lo, 2008]. However, there are limitations with hybrid systems, including handover overheads (when peers shift from one region to another); slow multi-hop connections and perhaps the most significant: an ever present (albeit reduced) dependency on central servers.

2.1 The general P2P approach

Systems designed on P2P principles are highly prized for their resource scalability, architectural flexibility and failure resistance. While traditional Client-Server (CS) architectures emphasize strict hierarchical structures, the P2P paradigm is more pliable and treats each network node equally. In situations where nodes are highly dynamic and heterogeneous, such as DVEs, this proves an ideal fit. However, there are fundamental challenges when implementing P2P-VEs, including:

- **Application/System Responsiveness** - DVEs, and games in particular, have an almost real-time constraint when it comes to communications. This ensures players experience a smooth, seamless and immersive virtual world.
- **Game-play Security and Consistency** - Cheat resistance is essential. Players will leave if a game or VE is perceived as easily corrupted, inconsistent and unfair.
- **Network Traffic Scalability** - The prevailing need is the minimization of bandwidth utilization, as unoptimized P2P systems generally incur far more traffic costs.

P2P networks are inherently decentralised, making management and control a difficult and complex task. Cohesion and consistency are put at risk in naive implementations and communication overheads could grow exponentially [Bharambe et al., 2008, Hu et al., 2006]. Thus, there is an almost paradoxical need to impose some limited structure, to securely support real-time game-play.

We extend the basics of Hu *et al.* [Hu et al., 2006], who utilized 2D Voronoi Diagrams to logically cluster peers and address network *scalability* issues. Our technique adapts 3D Voronoi Diagrams and uses non-spatial metrics for the newly introduced Z-axis. The goal being to perform better peer-clustering and aid in the selection of on-the-fly game-play arbitrators. Finally, our selection process employs varied unique mechanisms to speedily select them from localised, dynamic pools of peers.

By being decentralised, autonomous and flexible, our cumulative approach touches on all three P2P-VE concerns. Dynamic arbitrators assuages the *consistency and security* concerns, providing fairness in peer interactions. Finally, our selection timeliness addresses the issue of *responsiveness*. The importance of timely selection lies in the need to avoid multiple arbitration requests. Should an initial request fail for any reason, peers would need to re-negotiate for services. This is detrimental to game-play flow for two reasons:

- There will be inevitable extra processing and communication overheads, incurred with the subsequent requests.
- More significantly, the actual *start* of interaction (e.g., combat in a shooting game) is delayed.

For the second reason above, this is akin to asking a player in a typical First Person Shooter (FPS) game to *not* fire at an encroaching enemy. He must *wait* until we have arranged for someone to see his shot and resolve the resultant hit or miss. Obviously, this

is an unworkable mechanic in real-time interactive VEs, let alone typical fast-paced action games.

Our earliest forays into this research area [Almashor et al., 2010, Almashor and Khalil, 2010a], explore the use of Voronoi Diagrams (VD) for P2P-VEs and are the foundations to address these pressing issues. Voronoi Diagrams are fundamental geometric constructs [Shewchuk, 2002] used primarily in spatial partitioning problems. Standard 2D version provide an elegant peer clustering mechanism, sub-setting the peers into dynamic communication groups that minimize bandwidth.

This is in contrast to the approach used in [Bharambe et al., 2008], where each peer is expected to link to every other node in the network. More importantly, we inherit the strong cohesiveness and system-wide connectivity features of 2D-VD.

2.2 Peer-to-Peer Virtual Environments

While general applications such as email and video streaming remain essential Internet uses, computer games and virtual environments represent a growing facet of our online lives. In pure economic terms, the interactive entertainment industry generates \$65 billion in global yearly sales [Kushner, 2011, Baker, 2011], with predictions of even further growth. Simply put, more people are spending more time playing games and, most importantly, they are increasingly *doing it online* [Claypool and Claypool, 2006, ESA, 2011].

Online games [Ross, 2009, Achterbosch et al., 2007], have their roots in Distributed Virtual Environments (DVE) [Kwok and Wong, 2008, Matijasevic et al., 2002], which broadly refers to the simulation of interactive and immersive worlds over a communications network. They can mimic the reality around us or, as is often the case with the interactive entertainment medium, can represent an alternate fantasy setting.

The term DVE is itself an extension of the earlier Networked VE (NVE) [Macedonia and Zyda, 1997]. Here, the former term is preferred as it strongly implies a VE that is

distinctly partitioned across the network. This subtle nuance is doubly important in light of our aforementioned aim to enable a fully distributed VE over a P2P architecture. In such a system, each participating peer would also contribute processing, bandwidth and storage resources into it.

Other derivatives of DVE/NVEs are present in a myriad of other fields, such as the military [Macedonia, 2002, Zyda et al., 2003], social networking [Macedonia, 2007, Messinger et al., 2009] and collaborative tools [Boukerche et al., 2010, Rhee et al., 2007]. Again, the core idea is of a *shared* virtual world in which players interact with remote peers over the network.

Fig. 2.1 illustrates this concept. We see how a player is able to see and interact with his fellow players via his PC and a network connection. The architecture seen here, which is industry standard, follows a strict client-server (CS) paradigm. The player's actions and movement (and those of all others) are relayed to central servers, which essentially perform the role of communications middlemen.

This centralized connection model, with minimal use of direct inter-peer links, is the main point of contention. Key criticisms include the obvious networking and processing scalability issues on the side of the server, as well as its exposure as a single point of failure. Furthermore, players become trapped in distinct game instances and are cannot interact with players on *other* servers. Indeed, players are effectively partitioned into separate, independent “bubbles” of the virtual world, even if they are all playing the exact same game.

The CS paradigm is can thus be described as a limited number of clients consuming the resources of a single *isolated* server. Users access the game-world via home computers, which may be generalised as simple consoles transmitting user-input to an authoritative server. The server, in turn, processes all events and interactions, arbitrates over any conflicts before finally updating players on the changed state of the virtual world. Players can only see

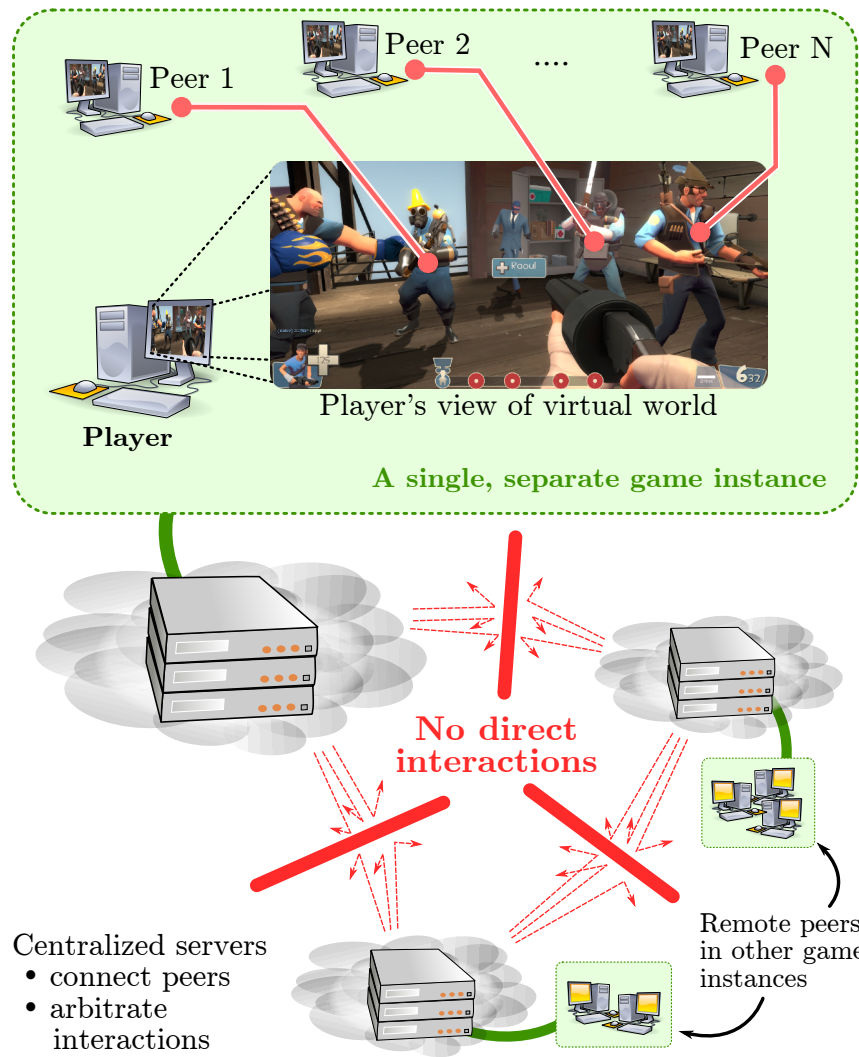


Figure 2.1: Typical connection and game-play characteristics of online games.

others within the confines of their connected server, and are at the mercy of the reliability of that single network entity.

The concept of a fully P2P-VE is an attempt to overcome these constraints and is driven in part by prior work in [Yu and Vuong, 2005, Hu et al., 2006, Rueda et al., 2008]. It strongly defines an underlying network architecture that is fully decentralised with regards to bandwidth, processing, management and storage requirements. The aim is to acquire

the resource scalability and resiliency features of a P2P system and are to also seamlessly inter-connect each and every player within the virtual world.

While P2P can theoretically counter these oft-cited issues with CS [Skibinsky, 2005, Neumann et al., 2007], it introduces issues yet to be adequately addressed in both academia and industry. There remains a lack of proper frameworks to ensure *secure and responsive* game-play, two factors that are vital to any DVE built on a P2P maxim.

2.2.1 P2P in Games

Most prior art has focused on using P2P techniques to solve related problems such as server and game-world availability [Kwok and Wong, 2008, Knutsson et al., 2004, Merabti and Rhalibi, 2004] and the partitioning (and thus load-balancing) of data and game-states within the DVE [Steed and Angus, 2005, Rhalibi et al., 2006]. Of late, the use of in-game spatial coordinates to divide the virtual world into more manageable portions has been gaining popularity. The architectures proposed are often of a hybrid nature, with elements of both CS and P2P designs incorporated.

More exploratory work has been done with regards to the effect of network quality and latencies on user experience [Chen et al., 2009, Armitage, 2003]. Such works emphasize our focus on FPS, as this class of games have the strictest network latency requirements. Thus, a solution in this area would likely be sufficient for MOGs in general. Some have tackled the issue of cheat-resistance and overall game-play security in P2P environments [Webb et al., 2008, Corman et al., 2006, Goodman and Verbrugge, 2008, Liu and Lo, 2008], of which the work done by Baughman *et al.* in [Baughman et al., 2007], in particular, stands out. Their basic Lockstep algorithm demonstrates that secure and fair game-play resolution is possible within a pure P2P environment.

Our use of the term “security” should not be construed in its classical sense. Rather, we mean security as defined by Baughman *et al.* in [Baughman et al., 2007], where the goal is to ensure correct and fair resolution of game-play according to established game rules

and mechanics. 3D-VD can help by facilitating the appointment of *dynamic game-play arbitrators*. These arbitrators can be thought of as short-term adjudicators of interaction and 3D-VD can help ease the issue of their selection.

Responsiveness, within the scope of our work, refers largely to the timely delivery of update messages amongst the participating peers. These updates drive the interaction and game-play within the VE and is vital to the user's game experience [Chen et al., 2009, Armitage, 2003]. Yet, this criteria must also be tempered with the need to limit bandwidth usage at each peer (and thus, the system). This is the working context of this paper: reducing network traffic while still maintaining timely updates in a P2P-VE.

The lack of any viable framework satisfying the above-mentioned criteria is the main barrier to the use of P2P in the games industry. Moreover, it also underlines the separation between DVEs from other types of P2P research. Distributed Hash Tables (DHT), distributed file-systems/databases and file-sharing systems [Rowstron and Druschel, 2001, Guo et al., 2007], are well-known within the sphere of P2P research. However, they do not share the almost real-time responsiveness and cheat-resistance requirements. Complications such as latency mitigation and game-play cheating immediately come the fore when considering P2P-VEs.

2.3 First-Person Shooters

Presenting an additional layer of complexity is the *type* of online game that we target: First-Person Shooters (FPS). Games can be broadly categorized according to their game-play mechanics, which is the prime factor influencing their communications latency requirements. A typical FPS game emphasizes fast reaction times and accuracy on the part of the player. Consequently, this class of online games are the most demanding in terms of latency tolerances.

Table 2.1 lists the three most popular types of commercially available online games and

Table 2.1: Comparison of the most popular game-types (with commercially availability)

Type	Concurrent Players	Max Latency	Remarks	Refs
RTS	2 to 12	1000 ms	No global VE continuity (i.e., one-off multiplayer matches); Players control hundreds of individual units; Some P2P mechanisms.	[Claypool, 2005, Claypool and Claypool, 2006]
MMO RPG/Sports	100 to 100,000+	500 ms	Persistent universe; Thousands connect to “shards” of the virtual-world; More relaxed game-play mechanics compared to FPS.	[Claypool and Claypool, 2006, Chen et al., 2009]
FPS	16 to 128	180 ms	No persistency requisite (one-off multiplayer rounds); Most stringent latency requirements due to reaction-based game-play mechanics	[Claypool and Claypool, 2006, Knutsson et al., 2004, Armitage, 2003, Quax et al., 2004]

their inherent differences. Massively Multiplayer Online Role-Playing Games (MMORPG) may scale to hundreds of thousands of concurrent players, due to more relaxed game-play and lenient aiming requisites. Subsequently, players have higher tolerances with regards to network latencies [Chen et al., 2009].

Real Time Strategy (RTS) and Sports games are unique in that players typically engage a limited number of opponents (for one-off online play). Like MMO-RPGs, game-play is not generally affected by speed and aiming accuracy, resulting in more relaxed latency demands (excepting racing games [Claypool and Claypool, 2006]).

In contrast, higher latencies and any resultant aiming inaccuracies would greatly affect FPS gamers. Simply, a player needs to know where his opponent is and he needs to know in a timely manner. Thus, if updates containing his opponent’s location are delayed, he will be greatly disadvantaged. Conversely, the sheer amount of bandwidth needed for high frequency updates forces developers to artificially limit the number of concurrent players per game server.

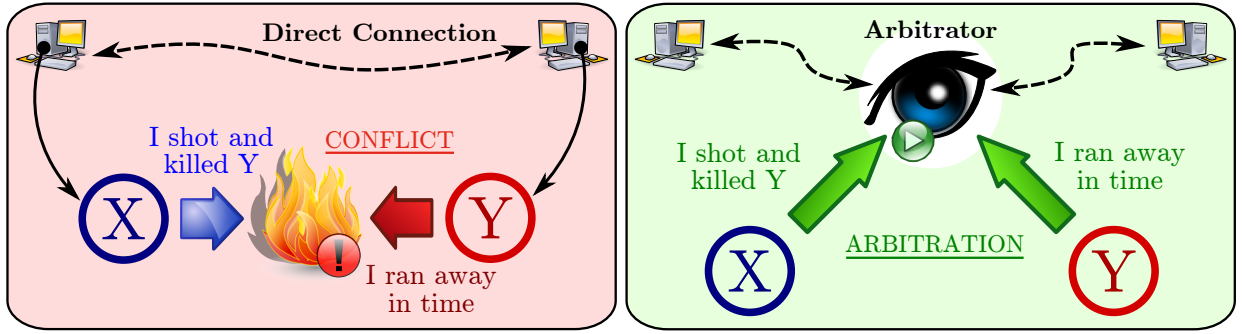


Figure 2.2: Illustrates the inherent conflict of interest when deciding game outcomes and the role of arbitrators in securing game-play fidelity

Of course, if we were to ask any gaming enthusiast for their preferred connection latency, the typical response would be “as low as possible”. However, the nature of FPS gaming places far more stringent requirements on the system. Thus, we can surmise that if we were to adequately serve the needs of the FPS class of games, we could potentially address issues in all others.

Ultimately, our efforts here strive towards a seemingly impossible task: an FPS-type game that is persistent and able to scale to extremely large numbers of concurrent players *in a single instance*. To the best of our knowledge, this concept (an MMO-FPS, if you will) has not been accomplished in either industry and academia and would be an exciting advancement in VE research.

2.4 Dynamic Game-play Arbitrators

Our focus on latency and responsiveness should not be construed as a direct effort to overcome the laws of physics and somehow enable speedier inter-peer links. Update messages will inevitably be delayed, either due to temporary lag spikes or persistently slow connections. This variability in the underlying network negatively impacts game-play, which can only be addressed with lag-mitigation techniques [Sweeney, 1999, Bernier, 2001] and the

use of *dynamically appointed arbitrators*.

Responsiveness in the interaction amongst peers is influenced by the speed of their individual connections. If we were to slow down the frenetic activity in games, we can generalise interactions to *tick-by-tick* events. Players move or perform an action and this is propagated to the server. In turn, servers process all actions, decide on the outcomes and subsequently updates all the peers.

Thus, a peer with a slow link will impact on the responsiveness of all other peers, simply by making an arbitrator wait for its updates. This is where lag-mitigation techniques come into play. They aid arbitrators in extrapolating the actions of slower peers, so as to not penalise peers with faster connections.

As seen in Fig. 2.2, game-play arbitrators also maintain fairness by being the adjudicators in any disputes. On the left, we see players X and Y with conflicting versions of events (and thus, conflicting data in their update packets). We cannot assume 100% fidelity as (i) either player may be cheating by way of packet modification; (ii) one or both are suffering slow connections. Thus, an independent fellow peer is brought in to intervene and decide on the outcome.

In current games, arbitration duties are performed by the *static* per-instance central servers (as seen in Fig. 2.1). They are the final authority on any in-game interaction. Effectively, peers commit all game-play actions (movement, shooting, commerce, etc.) to the server, which resolves game-play in accordance with established game rules. The burden of processing and state dissemination lies solely with that server, for all its attached peers.

In effect, peers replicate the banal tasks of simple input terminals, simply relaying commands from human players. Servers, being authoritative, are able to descry any anomalous, possibly malicious behaviour. For example, by detecting that a player has moved an impossible distance between consecutive turns or detecting corrupt or tampered update packets from participants.

Of course, with the move to large-scale fully P2P-VEs, the rigid centrally-managed structures used in current FPS games are wholly unsuitable. Yet, the vital function performed by these highly-resourced static servers must be replicated. Pursuant to this goal, the work here attempts to address the core issue of the *timely selection* of these needed arbitrators. Thus,

- In an architecture where all peers are deemed equal, how do we select suitable candidates from amongst the peer population?
- More crucially, to minimize any negative impact on the responsiveness of interactions, how do we perform this selection in a timely manner?

Moreover, the selection process needs to be done in a completely decentralised manner, in keeping with the concept of a fully P2P-VE. It has to curtail any latencies introduced during the process and minimize any impediments to the flow of a game. In essence, the work contained herein strives to *choose the right candidate first*, as we can ill-afford failed arbitration attempts.

The importance of timely selection lies in the need to avoid multiple arbitration requests. Should an initial request fail for any reason, peers would need to re-negotiate for services. This is detrimental for two reasons:

- The obvious processing and communication overheads incurred as peers seek new candidates to fulfil their arbitration needs.
- The inevitable delay in the actual *start* of interaction.

Elaborating on the second point above, this is akin to asking a player in a typical FPS game to *not* fire at an encroaching enemy. He must *wait* until we have arranged for someone to see his shot and resolve the resultant hit or miss. Clearly, this would be an unworkable mechanic in any fast-paced game.

Group voting and result validation techniques such as those proposed in [Baughman et al., 2007, Corman et al., 2006, Webb et al., 2008, Goodman and Verbrugge, 2008] incur greater overheads and are best avoided in light of our real-time constraints. We literally cannot afford to wait for each and every one of a small group of arbitrators to respond with its decision and to then tally the results.

Furthermore, these techniques are of greater complexity and rely on the assumed safety of group consensus. While more robust, the decisions of committees may not necessarily be more secure or trustworthy. Besides, how do we address deadlocked situations with equal number of votes for two outcomes?

Thus, the arbitration process can be further refined to the selection of a *singular* arbitrator in the shortest amount of time. Consequently, there is great emphasis on the criteria used during the selection process, such as:

- How do we filter suitable candidates from a large population of peers?
- Should we choose the nearest peers (for speedier set-up)? Or the furthest?
- What about peers with more available resources? How can they be selected?

Clearly, there are a myriad of considerations that can influence the appointment of arbitrators. Any proposed method would need to take into account (i) the large numbers of peers; (ii) the lack of any centrally managed architecture; (iii) the constraints of (almost) real-time FPS gaming; and (iv) the fairness and load-balancing requirements attached to arbitration duties.

2.5 Voronoi Diagrams in P2P VEs

We will not detail the exact techniques of creating 3D-VDs, rather we focus on using them purely as tools.

Two dimensional Voronoi Diagrams (2D-VD) have recently been introduced [Hu et al., 2006, Buyukkaya et al., 2009] as an elegant alternative approach. VDs are a fundamental geometric construct [Shewchuk, 1999], used primarily to solve spatial distribution problems. It is a method to partition any given area based on a given set of coordinates. Thus, in our context, a player's in-game positions are used to derive the VD. This ensures that peers are naturally clustered according to their locations (and by extension, their interests).

The primary use of 2D-VDs is to cluster peers into enclosed communication groups, minimising the bandwidth scalability issues inherent in pure P2P networks. By partitioning the peers, we reduce the amount of overall traffic while still maintaining VE cohesion and consistency. However, a 2D-VD enabled system is not without its own limitations. Game-play *responsiveness* and *security* remain the primary concerns that have yet to be adequately addressed. Our utilisation of *dynamic arbitrators* attempts to assuage these issues and thus enhance the application of VDs in the field of P2P-VEs.

Our approach extends upon the basic premise of 2D-VD by introducing a third dimension into the VD computations (3D-VD), enabling us to appoint game-play arbitrators from amongst the peer population. Depending on the metric used, our third dimension endows P2P-VE systems with features such as natural load-balancing and enhanced security mechanisms. There is also the ability to provide the lag-compensation techniques often employed by industry [Bernier, 2001, Sweeney, 1999].

2.5.1 Clustering Peers

The defining characteristic of prior attempts to introduce P2P features into on-line games and DVEs have been their attempts to cluster the peers into coherent groups. This is done primarily to minimise the bandwidth scalability issues inherent in pure P2P networks, where each peer is potentially expected to communicate with all other peers in the system in order to maintain VE cohesion and consistency. Thus, by partitioning the peers into communication groups, we reduce the amount of overall traffic.

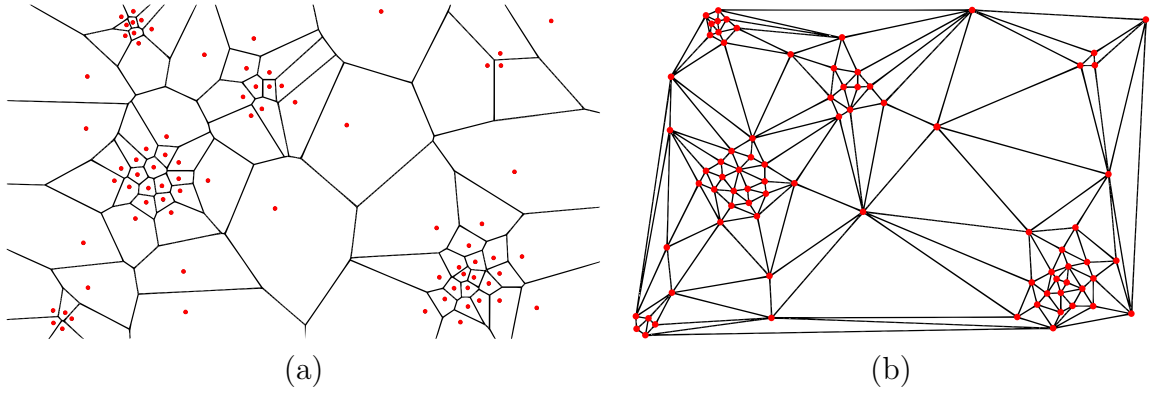


Figure 2.3: An illustrative 2D Voronoi Diagram, with its dual structure the Delaunay Triangulation.

This is where two-dimensional (2D) Voronoi Diagrams (VD), as seen in Figure 2.3a, have recently been proposed [Hu et al., 2006, 2008] as a novel mechanism to effectively group peers. The dots represent actual players within the network (i.e., a typical desktop PC which a human player interacts with) and the polygonal areas around them are their Voronoi cells. The illustration is a top-down perspective and we can see how certain areas of the map have a higher density of peers while others are much sparser. This reflects in-game interest and location mechanics as players often gravitate to each other based on their current activities (such as a fight or a meeting).

A VD and its accompanying dual structure, the Delaunay Triangulation (DT) as shown in Figure 2.3b, are fundamental geometric constructs [Shewchuk, 1999] utilised in the sciences to solve a variety of problems. In our context, it aids in the grouping of peers within the VE by means of their in-game spatial coordinates. Note how the DT representation shows inter-peer links. At the simplest level, a peer needs only to maintain connections with its closest neighbours, based on the DT.

In Fig. 2.3a, the dots represent actual peers within the network and the polygonal areas around them are their Voronoi cells. The illustration is a top-down perspective and we can see how certain areas of the map have a higher density of peers while others are

much sparser. This reflects the in-game interest and location mechanics as players often gravitate to each other based on their current activities (such as a fight or a meeting of friends). It must be noted that the way to obtain the VD representation is actually via the computation of a Delaunay Triangulation (DT), its dual geometric structure as seen in Fig. 2.3b.

The natural inclination is to use VDs to divide the game-world and then organise the peers based on the sub-partitions. This process is to be done dynamically by a central authority in response to changing network conditions, as in the approaches taken in [Buyukkaya et al., 2009, Hu et al., 2008]. In contrast, Hu *et al.* in their earlier paper [Hu et al., 2006] leverage on 2D-VDs derived *at each individual peer*, so that each one may find and connect with its nearest neighbours.

Using this technique, it is shown that the bandwidth scalability issue is suitably mitigated, especially when coupling the standard 2D-VDs with dynamic area of interest (AOI) management. However, while the traffic scalability issue is addressed, a myriad of issues remain which our novel use of the third dimension has the potential to solve.

Our initial experimentation with 3D-VD, as applied to P2P-VEs, suggests that there are valid avenues of research. We are concerned with extending the basic 2D-VD usage with a third dimension, in an effort to better cluster the peers. In particular, the aim is to supplement or possibly replace the use of in-game coordinates with other *non-positional metrics*. Most current research has concentrated on the use of strict, in-game positional data to derive their VD representations of the game-world. The approach outlined here paves a new direction by not only including the third dimension, but also utilizing it in a novel way, as we shall explain in subsequent sections in this chapter.

2.6 Dynamic Arbitrators using the 3rd dimension

By using on-the-fly arbitrators selected with our 3D-VD extension, we help address the twin issues of game-play security and responsiveness, two fundamental features critical to user enjoyment, especially in a fully P2P system.

To further explain, game-play security is made more robust by employing impartial arbitrators to resolve interactions between opposing peers. We understand *security* as it is defined by Baughman *et al.* in [Baughman et al., 2007] and to a lesser extent by others in [Webb et al., 2008, Corman et al., 2006]. That is, we are concerned with the *correct and fair playout of game-play*, as opposed to more general security issues such as intruder detection and identity theft.

The responsiveness issue is brought to the fore due to the physical constraints of network communications. One critical drawback of the approach by Hu *et al.* in [Hu et al., 2006] is that their algorithm assumes a direct, single-hop connection between peers will equate to timely delivery of update packets. This, under typical network conditions, is an unrealistic assumption. Current generation MMOGs and especially First-Person Shooters (FPS), address this latency issue via the use of various lag-compensation techniques, including (i) movement extrapolation and (ii) interpolation using prior updates [Bernier, 2001]. Such techniques require an arbitrator to act as an intermediary between peers, a requirement well-served by using 3D-VD.

Naturally, the inclination is to employ a central authority to partition the virtual world and thus organise the peers, as in the approaches taken in [Buyukkaya et al., 2009, Hu et al., 2008]. In contrast, Hu *et al.* in their earlier paper [Hu et al., 2006] leverage on 2D-VDs derived *at each individual peer*, so that each one may find and maintain contact with its nearest neighbours. Using this technique, they demonstrate that the bandwidth scalability issue can be addressed, especially when coupling VDs with dynamic area of interest (AOI) management. It is this work that our contributions are largely based upon.

2.6.1 Feasibility and Construction

The first step of our approach is to ascertain the feasibility of using three dimensional VDs. The computational cost of deriving VDs is an important consideration, especially when we take into account the latency-sensitivity of DVEs. Of particular note are First Person Shooter (FPS) games, for whom it can be surmised to have the strictest network latency requirements of any and all types of VEs [Chen et al., 2009, Armitage, 2003]. Game-play responsiveness is vital to the sense of immersion players feel within the virtual world and a protocol should seek to minimize any negative impacts to it.

2D representations may be computed in a timely manner, up to the order of thousands of peers [Hu et al., 2006]. However, as is the case with most computational geometry problems, deriving higher order VDs becomes progressively harder and computationally intensive with each additional dimension added. Each individual peer within the VE needs an accurate VD representation of the space around it and lengthy computation will seriously

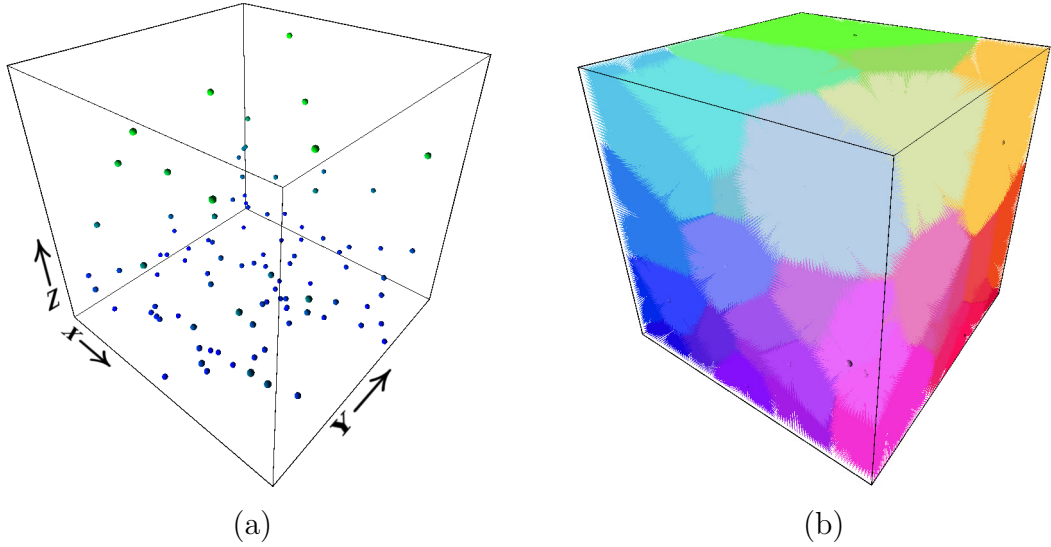


Figure 2.4: Extending from 2D Voronoi Diagrams: **(a)** reveals the 3D aspect by exposing the Z -axis; **(c)** demonstrates the resultant partitioning when the actual 3D Voronoi Diagram is derived from the peer positions.

impact game-play.

As can be seen in Fig. 2.4(a), the 3rd dimension can also be referred to as the Z -axis, with the corresponding X and Y axes already used to represent a peer's in-game location. Introducing it has implications on the complexity of the algorithm design and the resulting solution representations. Fig. 2.4(b) is the 3D-VD representation of the same population of peers, computed at an (now unseen) peer in the middle of the cube. We see how the Voronoi regions at the higher end of the Z -axis are generally larger when compared to the ones in the lower ranges. This is a function of the density of the peers at any one part of the cube.

The scale of the Z -axis also plays a significant role in determining the resultant DT. For example, a DT derived when the Z -axis is within the unit range of 0 to 100 is quite different to one obtained when the Z values are from 0 to 1000. The obtained tetrahedrons are different and as a consequence, so are the subsequent VD. We address this issue by standardizing the Z -axis with minimum and maximum values and having peers calculate their Z values as a percentage of the theoretical maximum.

As a rudimentary example, we can define that a dedicated game-server has the ability to simultaneously maintain 256 communication channels. Other normal peers may have half of this or even a quarter of this capacity. Thus, we take the value of 256 as the upper limit for the Z -axis, with other peers positioned according to their normalized percentages. In this scenario, a fully available server (with no channels used up), will be placed at the very top of the cube in Fig. 2.4(a).

We approached the feasibility issue by developing a software prototype which leverages on the open-sourced Computational Geometry Algorithms Library (CGAL) [Pion and Teilaud, 2009]. The testbed PC was a Linux desktop running an Intel Core 2 Duo processor (four cores @ 3000Hz) with 4GB of RAM. Due to the nature of CGAL, C++ was the development language.

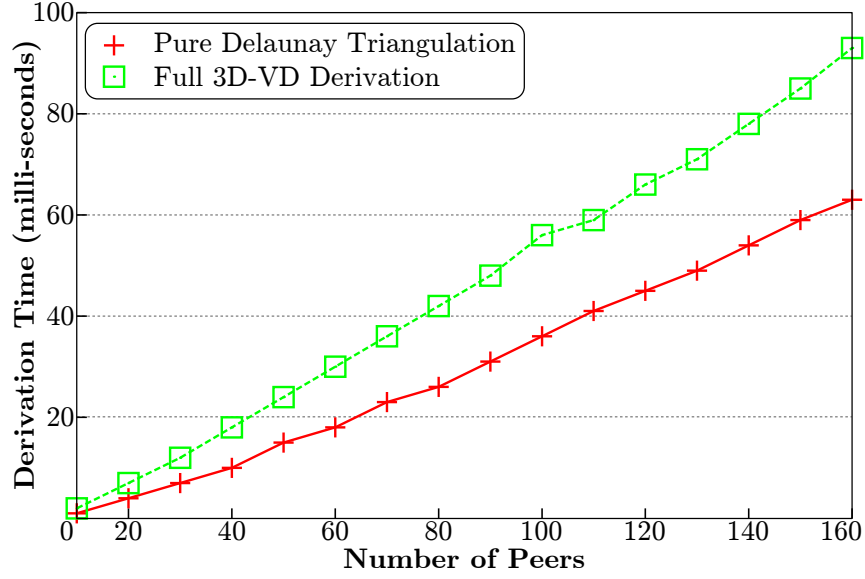


Figure 2.5: 3D-VD derivation times

Fig. 2.5 presents the results of our feasibility experiment. All derivation was conducted over multiple runs with the values shown being average time in milliseconds. The obtained results indicate that 3D-VD derivation is possible within normal game-play constraints. As a theoretical upper-bound, we use the industry standard of 30 frames per second (FPS) to render an image onto a user's screen. Thus, a maximum value of 33 milliseconds becomes the cut-off time for any derivation processing (shown by the horizontal line).

More importantly, the graph shows the increasing computation time of: (i) just the Delaunay triangulation (the lower plot) and (ii) the increasing time it takes to derive the DT and subsequently process the resultant 3D-VD (higher plot). We can see the overheads incurred with 3D-VD, which grows in a linear fashion and reaches the 33 milliseconds limit at around 60 peers.

The graph in Fig. 2.5 offers a more detailed breakdown of the costs involved when deriving and post-processing a 3D-VD. The neighbour search (minimising the list of fellow peers that a particular peer must connect to) remains fairly constant. This is a virtue of

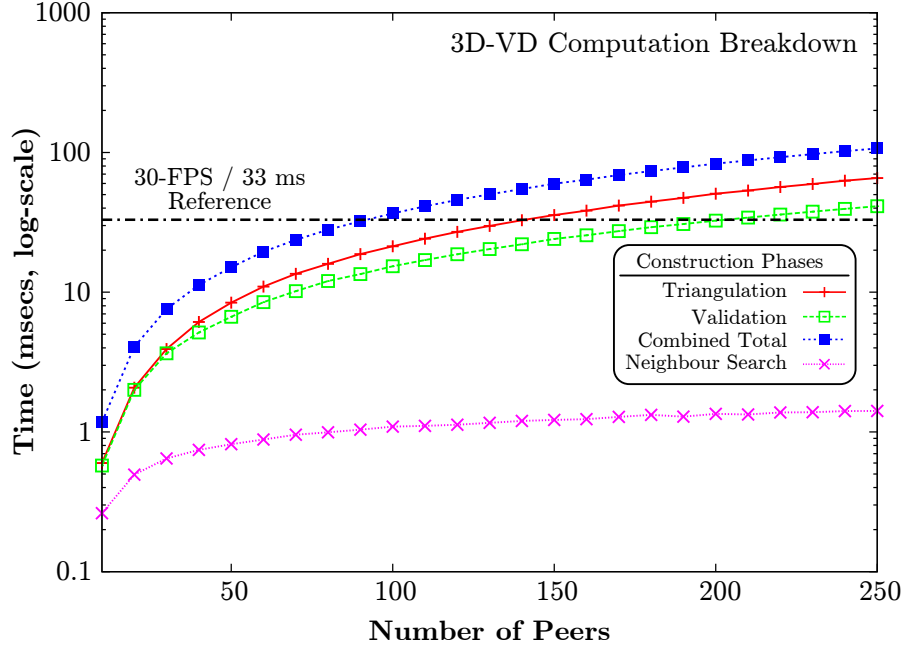


Figure 2.6: Breakdown of 3D-VD computation

using Voronoi diagrams as no matter how dense a selected area gets, the 3D-VD regions envelop around the computing peer and attenuates linkage requirements. In contrast, the bulk of the computation time lies with the purely geometric functions (i.e., performing the delaunay triangulation and validating it).

2.6.2 Computational and Connection Complexity

It must be noted that 3D-VD does increase network connections needed for each peer. With 2D-VD, a peer connects to a small subset of its neighbouring peers. However, with the increased exposure from the Z -axis, it is induced to establish more links. Moreover, there is increased computational complexity as we move from 2D-VD to 3D-VD, leading to slower derivation speeds that may impact application performance.

In light of this, it must be emphasized that 3D-VD works at a local level. Conceptually, there is no omniscient server constructing a 3D-VD for all peers. Rather, a 3D-VD is

computed by each peer for all of its neighbours, in order to discover new ones and determine who it should maintain direct links with, similar to the approach in [Hu et al., 2006].

Fig. 2.5 demonstrates the feasibility of using 3D-VD. The computation of the underlying Delaunay triangulation (a prerequisite, dual structure of Voronoi diagrams [Shewchuk, 1999]) is shown as the lower plot, while the full 3D-VD derivation (with neighbour pruning) is shown above it. While it is orders of magnitude more compute-intensive than simpler 2D varieties, we are still able to derive 3D-VDs within appropriate time-frames. Computations involving up to 160 peers are still performed within 100 ms, with the expectation that this will improve home-PCs become increasingly capable.

Thus, the computational complexity issues surrounding the creation and use of 3D-VDs, as well as network scalability concerns as the peer population eventually scales higher, are sufficiently addressed. To reiterate, the creation of the 3D-VD and subsequent clustering is *localised*. The sum of these individual clusters is what forms an extremely dynamic P2P overlay; one which is nimble enough to accommodate the frenetic pace and strict latency demands of typical FPS games.

2.7 Experimentation Details

All experiments reported within were performed with a specially developed simulation engine. The reason being the specific need for CGAL [Pion and Teillaud, 2009], an open-sourced, academic-driven C++ software library for advanced computational geometry. The lack of alternative software libraries in C++ (or any other languages for the matter), contributed to this decision, along with accessibility in terms of software platform requirements.

While other, more refined simulation engines such as Matlab and graphical/game libraries were considered, they were ultimately rejected due to a combination of cost, availability, and lack of functionality. Indeed, legacy games such as Doom (a popular FPS title from 1993, which has subsequently been made open-source), and its inbuilt network com-

munications code, could have been utilised. However, doing so would serve only to detract from the goal of exploring the usability and applicability of 3D-VD.

CGAL was chosen due to its performance and flexibility when deriving 3D-VDs. The libraries that took in a series of generated points, and then performed the actual 3D-VD construction and validation of its correctness, were linked statically to our own infrastructure code. We claim no credit for the creation and any modification of the geometric constructs. The Voronoi/Delaunay computation engine may be seen as a processing black-box within our context. Vertices defining the locations of individual peers within the virtual world were fed into the CGAL engine and we subsequently updated our own data-structures with the resultant values. This happens at every simulation step, with the process flow essentially being:

1. Gather peer positions along all three axes
2. Call the appropriate CGAL methods to generate a new 3D-VD
3. Update internal data-structures with updated coordinates
4. Apply experimentation logic onto updated data

Of course, significant work was put into the code surrounding the CGAL API calls. Maintenance and validation logic, as well as supporting infrastructure, was needed to ensure consistent and bug-free execution. As CGAL was written in C++, our own encapsulating simulation engine was necessarily developed in the same language. At the time, QT 4.6 and QTCreator was used as a development framework, due to its accessibility, relative ease of use, and its availability on multiple platforms (especially on our chosen Linux platform).

The hardware test-bed for all simulation runs were performed on a typical desktop PC with a four-core processor running at 3000Hz and 4GB of RAM. While not the most powerful, this test-bed served as good approximation of the typical hardware for FPS game

players at the time. It must be noted that the geometric computations were done on the CPU, which is a general purpose processor as opposed to dedicated hardware such as Graphical Processor Units (GPU) from the likes of Nvidia or AMD. As such, the derivation of 3D-VDs were necessarily slower as compared to being able to perform such operations on the parallel geometric rendering pipelines available on GPUs.

This limitation was imposed by our use of CGAL, as it was designed for CPU execution, and as massive parallelism via CUDA or OpenCL (in both hardware and software) were not mature at the time. This also constrained our simulation methodology, with the numbers of nodes necessarily capped to 2000, simply to aid in processing speed. It is conceivable that with the current state of on-demand cloud services and the enhanced functionality of GPGPU solutions (General-Purpose computing on Graphics Processing Units), that we can extend to a much larger scale in terms of experimentation.

However, since our mandate was the exploration and feasibility of 3D-VD, the relatively low numbers we tested on were deemed adequate to model both the frenetic activity seen in games, and the network bandwidth requirements that arise from such scenarios. Indeed, it is a reasonable argument that 1000-2000 peers would represent a microcosm of a larger virtual world, and that the fully decentralised and proximity-constrained nature of our technique would allow a fluid transition in scale (as explained in the precursor too our own work by Hu *et al* in [Hu et al., 2006]). It must also be noted the basis of our presentation is founded on prior art that performed experimentations on similar node population sizes.

Validation on the performance of our simulation engine was done at various levels. Early on in the experimentation phase, as we explored the creation of 3D-VD, the emphasis was on accuracy with low node populations. As an example, only 10 nodes were generated and their coordinates fed into the CGAL computation functions. The resultant geometric constructs were rendered onto an OpenGL canvas, with the help of QT modules that utilised underlying GPU hardware. This allowed an interactive visual representation of

the DT or 3D-VD, where we could manually inspect the step-by-step movement of nodes. Indeed, some figures contained within, such as Fig. 4.3 were captured from such renderings (though they were subsequently used for illustrative purposes). This provided us assurances on the correctness of our engine and the performance of the CGAL method calls.

As the research progressed, we tested with increasing population sizes; from 100 peers, to 500, and finally the full simulations with 1000 and 2000 peers. At each step, we manually checked both the first few initial simulation steps, and the final end-results to be confident of the correctness of the engine. As we mostly chose a randomised movement template for peers (in all experiments save for the flocking-mitigation technique in chapter 5), full validation could only be performed after each simulation run; for example, by analysing compiled results on the performance of individual nodes with earlier records. In addition, pre-generated and fixed starting coordinates were used for all peers on all run-batches. This is, in effect, a fixed pseudo-seed that simulation runs were launched with and checked against. Fixed movement testing was also done, where we restricted the randomised peer movements to a certain direction and magnitude, in order to observe the performance of the engine.

Lastly, before each final simulation run (that produced the results seen later), multiple pre-trial runs were performed to ascertain the stability of both the engine and the end-results. Direct comparisons against prior techniques proved difficult, due to the novelty of both our approach (i.e., the use of 3D-VD) and the questions we seek to address (arbitrator selection and performance), and the fully decentralised context we worked in. Of course, we have provided theoretical validation against certain aspects of pre-existing techniques where appropriate.

Chapter 3

Timely Arbitrator Selection with 3D-VD

Our use of 3D Voronoi Diagrams follows a simple premise: take the standard 2D versions used in prior art [Hu et al., 2006] and introduce a 3rd dimension. Thereafter, we innovate further by using a *non-spatial metric* for the Z-axis. This is in an effort to address our second research question of the efficient application of 3D-VD onto a P2P-VE platform. The end result is a dynamic technique that can seamlessly connect an extreme number of peers, provides a flexible way to cluster them, minimizes wider system scalability issues and fluidly allows us to employ aforementioned client-server arbitration tools.

3.1 Working with 3D-VD

Voronoi Diagrams, as seen in Fig. 3.1, are fundamental geometric constructs [Shewchuk, 1999, 2002]. Their main utility is the resultant and natural spatial partitioning when they are applied to the virtual world. Simply put, given a list of coordinates (i.e., of the peers), VDs clearly divide any given space into zones of control for each coordinate. From this, meaningful peer relationships can be derived, allowing us to discriminate between peers based on the Voronoi zones.

For example, in a 2D map as seen to the left in Fig. 3.1, we can easily see when two peers may be of little interest to each other. Not because they are deemed too far by

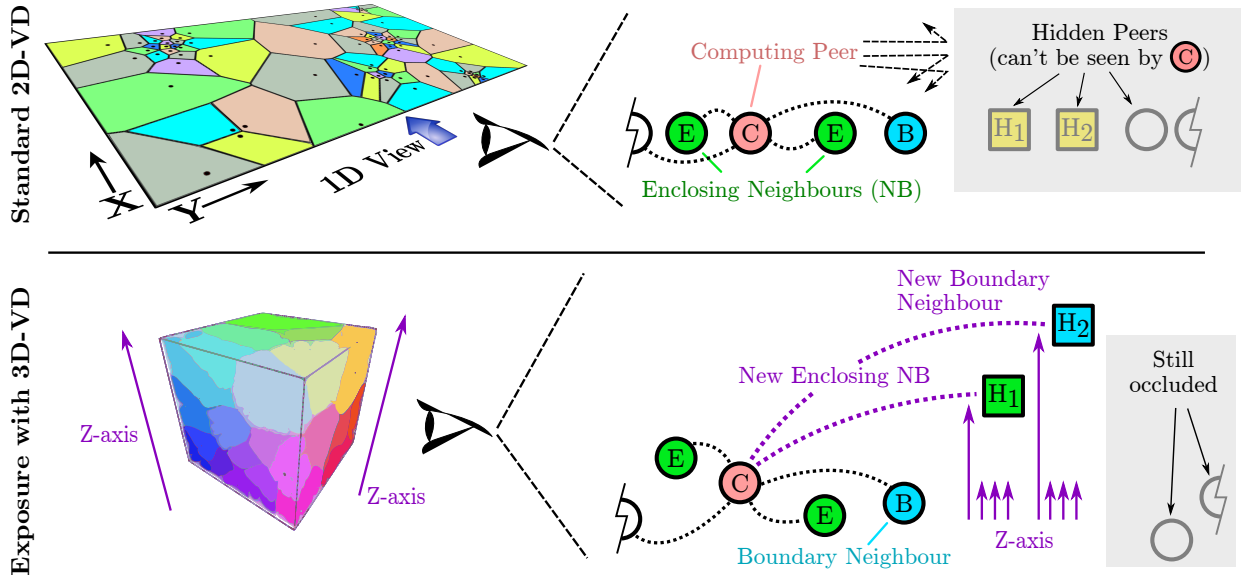


Figure 3.1: Illustrates the differing in-game view of peers within the P2P-VE when using 2D (top), as opposed to 3D Voronoi Diagrams (bottom). It is seen how the added exposure from the Z-axis allows the computing peer to see and link with previously hidden neighbours.

some measure; but because 2D-VD tells us there are intermediary zones (and by extension, peers that are of more interest). Thus, we need not depend on rigid and arbitrary distance benchmarks. The goal is to use the natural clustering tendencies of 2D and 3D-VDs, as noted in [Hu et al., 2006, Steiner and Biersack, 2006].

From a reciprocal standpoint, we see that a peer may be forced to maintain links with a distant fellow peer in 3D-VD, simply because of their adjacent zones. This is the key feature of that we utilize: the fact that even if two peers are separated by a great distance and/or many intermediary nodes, they could still theoretically be neighbours (NB) due to their Z-axis positions.

3.1.1 Neighbour Clustering

The idea is to cluster the peers in a dynamic and efficient way. The reasoning behind this is the fact that we cannot possibly provide a direct link from each peer *to every other peer*.

The bandwidth demands would be prohibitive in a P2P-VE with hundreds, thousands and perhaps, millions of players.

There are works that do attempt this, most notably in [Bharambe et al., 2008], with interest management and rate-limiting techniques used to mitigate bandwidth issues. However, we argue that there is little benefit to continuously update hundreds of possibly disinterested players situated on the far side of the virtual map.

Accordingly, the reader may wonder: How exactly does 3D-VD help us? How is it more beneficial over 2D ones? Referring back to Fig. 3.1, the 3rd dimension elegantly exposes, for the computing peer C , a number of previously occluded NBs. With 2D-VD (left), potential high-capacity NBs (H_1, H_2) are blocked from view by intermediary enclosing (E) and boundary (B) nodes.

On the right, we observe how instituting the Z-axis allows C a flexible way to “see” beyond its immediate NBs. We thus bypass the restrictive proximity constraints of a 2D map, allowing each peer to have natural and *direct* links to potentially more desirable NBs. We are in effect expanding each peer’s connectivity horizon but doing so in an elegant and controlled manner.

The terms “enclosing” and “boundary” were first defined in VON [Hu et al., 2006] as a means to categorize the world around a particular peer. If that central peer creates a Voronoi representation of its surrounding NBs, the most immediate ones would form an enveloping set of cells (polyhedral areas in 3D-VD as opposed to polygonal spaces in 2D-VD). These NBs are deemed as *enclosing*.

Subsequently, *boundary* NBs form yet another, larger envelope of Voronoi cells that completely encapsulates both the central peer and the cells of its enclosing NBs. All other peers that fall beyond these two types are ignored.

It is important to note that the VD computations are *performed at each peer*. That is, a peer calculates a Voronoi representation of the world around it and prunes its con-

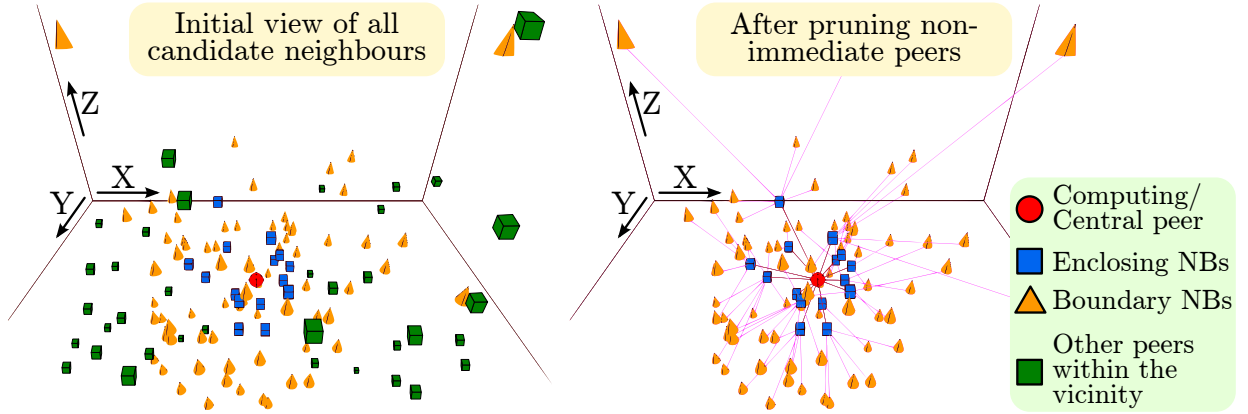


Figure 3.2: Showing the 3D version of the Voronoi pruning process adapted from [Hu et al., 2006]. *Note: The actual polyhedral Voronoi cells were removed to enable visualisation of the peers.*

nections accordingly. There is no central entity that takes in the locations of all peers, derives a 3D-VD for the entire VE and then assigns the neighbours for each peer. Doing so would defeat the purpose of having a fully P2P-VE, as we would need that central authority. Additionally, the time-cost of deriving a 3D-VD for millions of nodes would alone be prohibitive.

Fig. 3.2 exemplifies this process with a zoomed-in view of a central peer and its initial surroundings. At each tick, the peer receives knowledge of the others around it via mutual update packets. Armed with the X, Y, Z coordinates of its candidate neighbours, a 3D-VD is derived and then used to prune candidates down to a select few (as seen on the right of Fig. 3.2). This is repeated by every peer in the simulation, *at every tick*. This is the same methodology as employed in VON, except with the added Z -axis.

A more robust examination of the general technique, albeit for 2D-VDs, is found in [Hu et al., 2006]. Suffice it to say that our own extensions generally follows the 2D methodology. The main motivation here is to build upon the features of 2D-VD, augmenting it with a Z -axis to aid in the task of selecting arbitrators.

3.2 Metrics for the Z -axis

Using in-game coordinates provides a natural clustering mechanism, yet there are vital limitations. In the case of 2D-VDs, spatial relationships constrains the process by allowing only peers in close proximities to maintain direct links.

This approach is inadequate when considering that there may be two peers, P_A and P_B , who are spatially separated by a large distance but still retain an interest in one another. The 3rd dimension allows us another angle of approach to this problem. For example, the 3D-VD may easily be used to appoint a third peer as an arbitrator, P_{Arb} , that maintains a link between the two.

The question then becomes: What may be used as the independent measure for the 3rd dimension? This freedom to select a suitable metric endows added flexibility to the 3D-VD technique, enabling our mechanism to achieve varying goals dependent on the metric chosen. Some of the possibilities include:

1. **Individual Peer Reliabilities** - A measure of a user's expected lifetime within the VE, enabling selection of more stable arbitrators.
2. **Player's Cumulative Reputation** - May be used to judge the trustworthiness of an arbitrator, thus remaking 3D-VD into a security mechanism.
3. **Resource Capacity** - To better manage the use of resources and hardware limitations at each peer, effectively making 3D-VD a load-balancing tool.
4. **In-game Activity Level** - Gauges the potential connections between peers and thus predict expected demand in areas of the virtual map.

A natural instinct would be to simply use a peer's in-game Z coordinates and directly translate it to the Voronoi space. For example, we could track the vertical location of

player's avatars and use their elevation within the game-world as a means to discriminate along the Z -axis in our 3D-VD computations.

Under more scrutiny however, we see the pitfalls of this approach. Firstly, most games (and especially FPS) are played on fairly flat virtual terrains. Even recent games set in urban environments and featuring aerial vehicles, combat is usually consigned close to the ground. Thus, the resultant 3D-VD derivations would not confer any real benefits over 2D-VD varieties.

Furthermore, a crippling flaw is found when we consider that a peer may, due to game-play reasons, rise to the top of the 3D-VD. Such would be the case if a player decides to climb a tower to better target an enemy. Consequently, there is then a fatal disconnect between (i) his suitability as an arbitrator and (ii) his Z -axis position: Is he more reputable just because he is vertically higher? Does he have more bandwidth to justify his high 3D-VD position?

3.2.1 Resource Capacity

To further our discussion, we chose **peer resource capacity** to augment the X and Y in-game coordinates. More specifically, we aim to use the ever-changing *current* resource capacities of peers to position them along the Z -axis. This is a simple but nevertheless important metric with which to differentiate peers, especially in a fully decentralised network environment.

Each peer has its own limitations and providing a way for the system to adjust to those constraints is a useful load-balancing feature. It also caters to the dynamism in P2P network topologies as a peer's changing situation can be quickly factored into the VD computations. The term resource could be taken to mean either bandwidth, processing, storage resources or even an aggregation of all three. We are inclined to use bandwidth capacity as our measure, as it is usually the most constrained with regards to DVEs and MMOGs [McCoy et al., 2005].

Significantly, using resource capacity as our metric allows the Z -axis to be entirely independent of the other axes and also independent of other peers. To explain, the X and Y coordinates of a player within the game-world has no bearing on his PC's resource capacity. This ensures a clean separation between the metrics and avoids potential peer-clumping effects. In other words, a player's position has no influence on the Z -axis when we compute the 3D-VD.

This independence also extends to the computing peer's relationships with other peers. Simply put, there is an entire lack of dependencies amongst peers for the 3rd dimension. A particular peer's bandwidth capacity is not measured against or in relation to other peers (with the sole exception of the predetermined theoretical maximum). For example, if we were to use the ping time from one peer to another, the metric would be a *relative one*. It would not be suitable as the obtained values would be too disparate and cause unnecessary clumping of peers. Using independent metrics assuages this issue.

The other candidates, Individual Peer Reliabilities and In-game Activity Level, were deemed too susceptible to player behaviour and thus highly-volatile. Peer reliability aims to predict a peer's lifetime within the DVE and seemed, initially, to be a good metric. However, as noted in [Chen et al., 2009], there are various external (to the system) factors which affect a player's projected involvement. As an example, even with a high current reliability score, a peer may just disconnect due to network conditions or a player simply logging off.

Using a peer's in-game activity level is also dependent on player behaviour and, to a certain extent, on other peers. This also makes it less usable in our approach. It must be said however, that these two metrics would make interesting studies and could indeed be introduced in future work.

3.2.2 Peer Reputations

Peer reputation is an important metric, but one that shifts the focus of our approach to address the security aspects of game-play. Using peers with high reputations as arbitrators would be a desired logical feature in any P2P-VE.

However, peer reputation constitutes a vast area of research, requiring a significant undertaking to correctly implement. Our own approach mirrors that of the games industry, where game-play responsiveness is (rightly) targeted first, with security a secondary aim. After all, a highly secure FPS game would be hard-pressed to attract players if it is also slow and unresponsive.

That said, it is certainly a promising metric and, at this time, remains part of future work. The intention is to substitute one of the in-game coordinates (X or Y) for peer reputation. This will further augment our base 3D-VD system and introduce an additional security feature by partially obfuscating player positions within the game-world. This approach is an effective compromise between the VON system in [Hu et al., 2006] and the LockStep protocol as defined in [Baughman et al., 2007].

3.2.3 Scale of Z -axis

There are slight complexities when translating the fluctuating resource capacities of peers onto the Z -axis. Crucially, the scale of Z -axis can alter the final 3D-VD representation. A 3D-VD derived when the Z -axis is within the range of 0 to 100 is quite different to one with Z values from 0 - 1000. The resultant polyhedral Voronoi cells are squeezed or stretched accordingly and thus, would lead to very different 3D-VDs and any derived inter-peer linkages.

This is addressed by fixing the minimum and maximum values of the Z -axis and having peers calculate their positions as a percentage of the maximum. As an example, we can define that a server has the ability to simultaneously maintain 256 communication channels

and this is made the global maximum Z value. Accordingly, super-nodes can only reach half of this value, at 128; Normal peers may reach 25% (64); and weak-nodes can only go up to 32 (12.5%). Within the context of the 3D-VD, a fully unloaded server (ie., with absolutely no arbitration duties) would float at the very top.

3.3 Selection Policies

With our emphasis on responsiveness, we reiterate that the goal is not to select quick arbitrators. Rather, we want to *quickly select* arbitrators. Improving actual connection speed between arbitrators and peers is beyond the scope of the work here. However, this issue is indirectly addressed by using upload capacities (currently the most limiting resource [McCoy et al., 2005, Bharambe et al., 2008]) as the metric for the Z -axis.

3.3.1 Timely Selection

The selection mechanism is the cardinal concern here, with the whole process ideally taking the least amount of time possible. Time, within our context, is defined as discrete and uniform simulation *ticks*, rather than actual milliseconds. We can then distil more meaningful data and analysis from the simulation engine and preclude the variability of real-world connections.

Our goal is that, once a peer requests for arbitration in one tick, it ideally should receive it at *the very next tick*. For example, a peer moves around freely from ticks 10 to 20 (t_{10-20}). At t_{21} , he spots an adversary (another peer) whom he then decides to attack. This is when the selection process is triggered (i.e., at t_{21}) and ideally, an arbitrator is selected from his known neighbours at t_{22} .

Ill-informed choices when selecting arbitrators causes failures and subsequent re-negotiation which ultimately delays the start of interaction. Supposing a peer selects an already constrained node which then rejects the request, an entire tick is lost and extra costs are

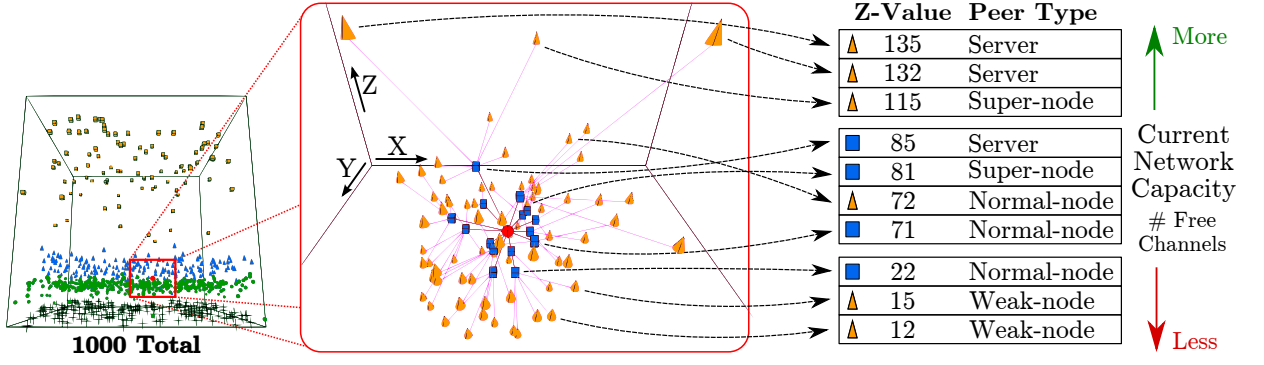


Figure 3.3: Illustration of how a peer derives its own 3D-VD representation of the others around it, decides on its neighbours (NB) and subsequently ranks them in terms of suitability. Nodes seen here were generated for illustrative purposes.

incurred to repeat the process.

For our simulations, only singular arbitrators are used, due to previously mentioned concerns with the complexities of group/voting consensus, inter-arbitrator latencies and additional overheads. Nonetheless, if so desired, 3D-VD can be easily modified for multiple selections. Furthermore, at this stage of our work, choosing safe and reputable peers is not yet a primary concern, although some of our selection policies do factor in security aspects.

Our use of the term “timely” is an abstract interpretation decoupled from the standard concept of time. Rather, we utilize the notion of “tries”, effectively measuring the number of attempts to obtain arbitration services. A peer may only make one attempt per simulation tick, making it imperative to minimize retries. Failed attempts delays the start of arbitration and thus impacts game responsiveness, with bandwidth wasted due to repeated requests.

Flooding mechanisms are to be avoided, due to overheads and the fact that three ticks are required: one to flood; one to receive replies; and the last to confirm and begin. In contrast, our approach only incurs costs when there are failures, with the onus being to “get it right the first time”. When coupled with a strong selection policy, we can circumvent the first two phases in flooding.

3.3.2 Candidate Policies

Fittest (FIT): A simplified policy whereby we utilize only knowledge derived from the Z-axis, where peers with more available resources are progressively placed higher. Fig. 3.3 aptly demonstrates this policy within the context of the overall 3D-VD mechanism. Effectively, the candidates (all of a peer’s NBs) are ranked according to the third dimension and the top-most is chosen.

Coincidentally, it is seen that FIT also facilitates the selection of the *most suitable* arbitrator, simply by appointing one with the most available upload bandwidth. It is a safe decision to constantly select the topmost Z-axis candidate, as it would likely be well-resourced and thus, not reject the request.

Random Arbitrator Selection (RAN): This is the baseline policy, where we select arbitrators randomly from the requester’s pool of NBs. Doing so allows a uniform spread of arbitration loads, as each candidate NB has an equal chance of selection. Further, RAN is arguably a security mechanism as well. As arbitrators are effectively appointed by chance, malicious peers cannot predict future arbitration selections and thus unfairly influence game-play.

Nearest (N2D/N3D): Here, responsiveness takes precedence over security. We choose the closest NBs as they are likely to already have localised player information (e.g, current health). Thus, we limit any pre-processing and data-exchange to the arbitrator, ultimately lowering the total set-up time.

This approach also builds on the idea that, players crowding an area in the virtual-world also tend to be in same real-world geographical region [Safaei et al., 2005]. Using euclidean distances on a 2D plane (ignoring the Z-axis), the closest NB is always chosen. Pursuant to this, we may infer that the network distance between peers and the candidate (and thus, their latencies) are minimised.

With 3D distances, the Z-axis component is also considered, which favours nearest NBs

that have approximately the same current load as the requester. So, if a peer is currently experiencing medium traffic (e.g., mid-height in the cube in Fig. 3.2), it is likely to choose NBs from the same layer. The idea is to choose candidates will be of similar capability (peer-type) to the requester. A use case for this is if a system designer would want peers to maximise the use of their fellow (same capacity) peers, before seeking help “from above”.

Out-most (O2D/O3D): These form the counter argument to N2D/N3D, emphasizing fairness instead. When requesting arbitration, there is a need to select peers further away, as they are, ostensibly, uninterested in local events. The inference is that such NBs are less motivated to unfairly interfere with outcomes involving a far-away requester and his intended opponent.

A nearby candidate, on the other hand, may have a vested interest and can affect outcomes negatively, to potentially his own game. For example, a player, P_X , guards a valuable resource that a nearby arbitrator, P_A , desires. As P_X battles with an opponent P_O , P_A can maliciously resolve game-play outcomes in favour of P_O and thus easily access the valuables thereafter.

Central (C2D/C3D): This policy simply aims to strike a balance between prior distance-based policies, juggling fairness and responsiveness. Initially, all the NBs of the requesting peer are sorted according to their distances from it. From this sorted list, the median NB (i.e., the middle-most) is chosen, ensuring that a relative (as opposed to absolute) measure guides the process.

Mass Effect (MST/MDY): Here, inspiration was taken from the fundamental Newtonian laws on the gravitational pull of masses, and its roots in the widely used inverse square law. Within the context of our 3D domain, Newton’s model is ideally suited to exemplify peer relationships in the VE. Thus, calculating the force of attraction, F , between two peers, is given by:

$$F = G \frac{M_C M_N}{D^2} \quad (3.1)$$

where G is the gravitational constant; M_C represents mass of the computing peer; M_N is the mass of a NB of M_C ; and D is the distance between them. Our use of the term mass is an euphemism for capacity, specifically meaning a peer’s resource levels. Thus, the more resources it has, the more “massive” it is and the greater the strength of its gravitational pull.

We further classify two unique variations: Mass-effect Static (MST) and Dynamic (MDY). MST uses unchanging resource values, effectively locking a peer’s capacity. Thus, a dedicated server is consistently massive while a weak-node stays minute. In contrast, MDY embraces the fluidity of peer capacities. Both schemes seek to balance mass attraction principles (biasing towards large capacity NBs) with a strong distance decay factor that prefers nearer NBs.

To conclude, the above policies augment the 3D-VD mechanism and whilst some may seem simplified, the dynamics in the selection process are adequately modelled and are sufficient to test the validity of the proposed method.






3.4 Experimentation

For our experimentation, a specialized simulation engine was implemented. This was essentially due to the need for CGAL [Pion and Teillaud, 2009], an open-sourced, academic-driven C++ software library for advanced computational geometry. CGAL was chosen due to its performance and flexibility when deriving 3D-VDs.

3.4.1 Limitations

The need to test with large enough numbers of peers precluded any inter-client experimentation with actual network communications. Thus, simulations are in order, with most

Table 3.1: Details of simulated peer populations and their notations in later results.

Name	Abbrv	Plot	Description
Egalitarian	Ega		A standard, mixed population consisting only of normal-nodes and weak-nodes.
Supernode Enhanced	Enh-Sn		Minority of Super-nodes (10% of population) are used to bolster a mix of normal/weak nodes.
Server Enhanced	Enh-Sv		Similar to Enh-Sn but with dedicated, higher capacity servers (4 times that of a normal-node).
One Server	Sv1		Special case where a sole high-capacity server services an Egalitarian population.
Four Servers	Sv4		Similar to the sole-server configuration but with 4 high-capacity servers, instead of just 1.

tests conducted with a chosen peer population of 1000. This may seem small but we are presently bound, by the available hardware, to the computational and time constraints of

Table 3.2: Explanation of **Event Types** (as shown on the X-axis of all results graphs)

Abbrv	Explanation
Trs	The number of Tries represents the total interactions (and thus arbitration requests) generated by peers throughout the simulation.
F1	Fail-1 indicates the subset of Tries which subsequently failed. That is, at the first attempt seeking arbitration services, the request was rejected.
F2 F3	Fail-2/3 are further, consecutive failures as peers repeat their request for arbitration services during successive ticks (i.e., after the first fail event).
FG	After three continuous failures, peers give up, denoted by FG (Fail-G)
Lst	Lost means a loss of contact between two neighbouring peers. E.g., two peers wish to engage each other in combat and, due to a failed requests, they give up and move apart. This is another form of Fail-G , signifying peers who become disinterested after failing to secure arbitration services.

deriving the 3D-VDs. At each tick, a 3D-VD of all peers within the VE is constructed, which is a computationally intensive task that is then repeated for the entire run.

Accordingly, we cannot realistically compute 3D-VDs for millions of nodes, as it is presently beyond the computational power afforded to us. However, it is stressed that the results here can be reasonably extrapolated to larger numbers, as the tests speak to the validity of our approach on smaller subsets.

Peers are contained within a single simulation instance, with results collected per simulation “tick”. Each tick represents a discrete time-step where peers execute moves and resolve interactions, for a period of 100 ticks per simulation run. We assume no peers become disconnected, as this is not our primary focus here. Moreover, there is no need to duplicate prior experimentation in [Hu et al., 2006, Rueda et al., 2008] where effects of peer losses/delays are already well documented.

For simplicity and to speed up our simulations, our engine derives an all-encompassing 3D-VD per tick, which each peer subsequently uses to find its neighbours. In actual fact, each peer must compute its own 3D-VD representation of the virtual world, and use it for NB discovery. As there are no peer losses, subsets of larger 3D-VDs are equivalent to self-derived ones.

3.4.2 Simulation Set-up

Peer movement within the virtual map is randomised, in accordance with the simulation methodology of [Hu et al., 2006]. While it arguably represents the basic behaviour of players, we are currently implementing movement algorithms that better mimic the flocking/crowding characteristics seen in most games.

The population distributions of the four distinct peer types are summarised in Table 3.1. Configurations are meant to approximate peer distribution in any would-be MMO-FPS. “Sv1” and “Sv4” scenarios were conceived to measure the stresses placed on dedicated servers, if any are installed into the P2P-VE. Table 3.2 summarises the event types recorded

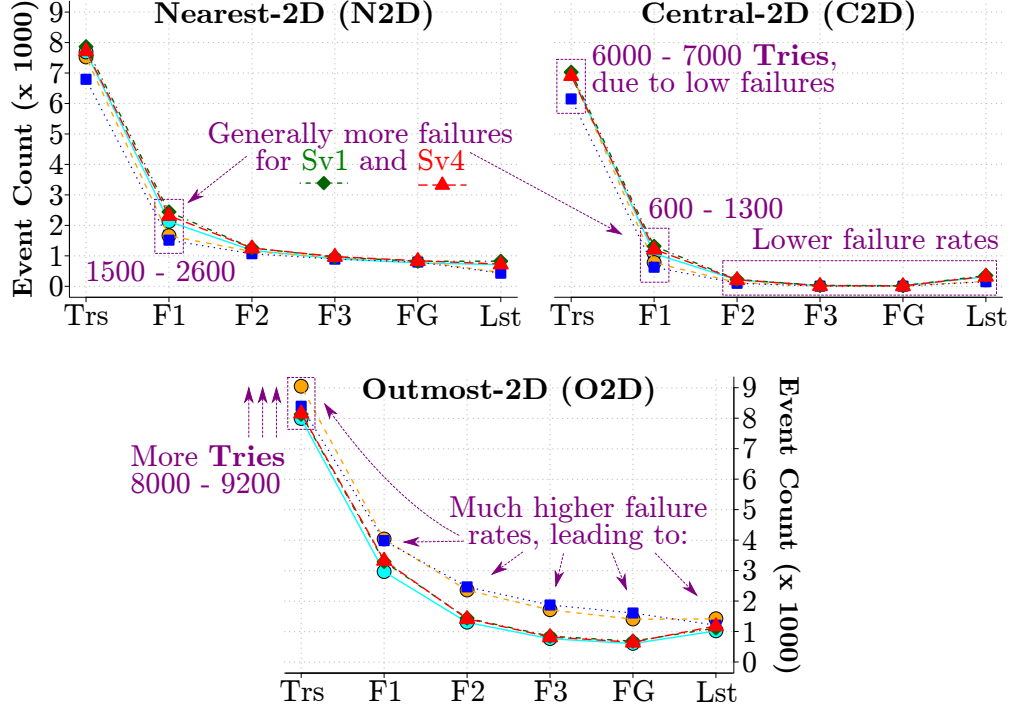


Figure 3.4: Performance of 2D Euclidean-distance policies. From top-left, clockwise: **Near-****est**, **Central** and **Outmost**. Their 3D counterparts are not shown as plots very closely follow the results here.

in the simulations. Failures denote rejection of any arbitration requests by the candidate peer. For example, the candidate receives some number of requests during a tick and decides that it is unable to accept any more arbitration duties, based on its present load. It then rejects the incoming requests with a simple reply back to the requesters.

3.4.3 Results

Turning to the actual results, we begin by examining the failures recorded for 2D distance-based policies as seen in Fig. 3.4. It is surmised from the plots that distance-based schemes exhibit average to low performance, with Outmost (O2D) especially poor (as evidenced by the elevated failure counts). Choosing centrally distant NBs (C2D – middle) limits failures to the low thousands and constitutes the best failure avoidance of the three “blinded” policies, where dynamic upload capacity is effectively disregarded as a selection criteria.

The results for 3D-based euclidean distance measures generally follow that of 2D. They represent our attempts to include current nodal capacities (embodied by the Z-axis) into the selection decision. Again, central-themed policies offered the best results, generally keeping failure rates below 1000.

Analysis of the mass-effect policies seen in Fig. 3.5a and 3.5b shows close tracking with the Nearest (N2D) policy. This is due to the inverse-square laws favouring nearer NBs even if they are less fit. Note the improvements when using dynamic over static policies, reflecting the importance of utilizing the fluidity of peer upload capacities in any selection mechanism.

In all the plots shown here, we also note the “snow-balling” effect of first-failures and the subsequent substantial rise in the number of **Tries**. This is most obvious in the O2D plots of Fig. 3.4. Increased failures cause peers to retry with *new* interactions, driving up

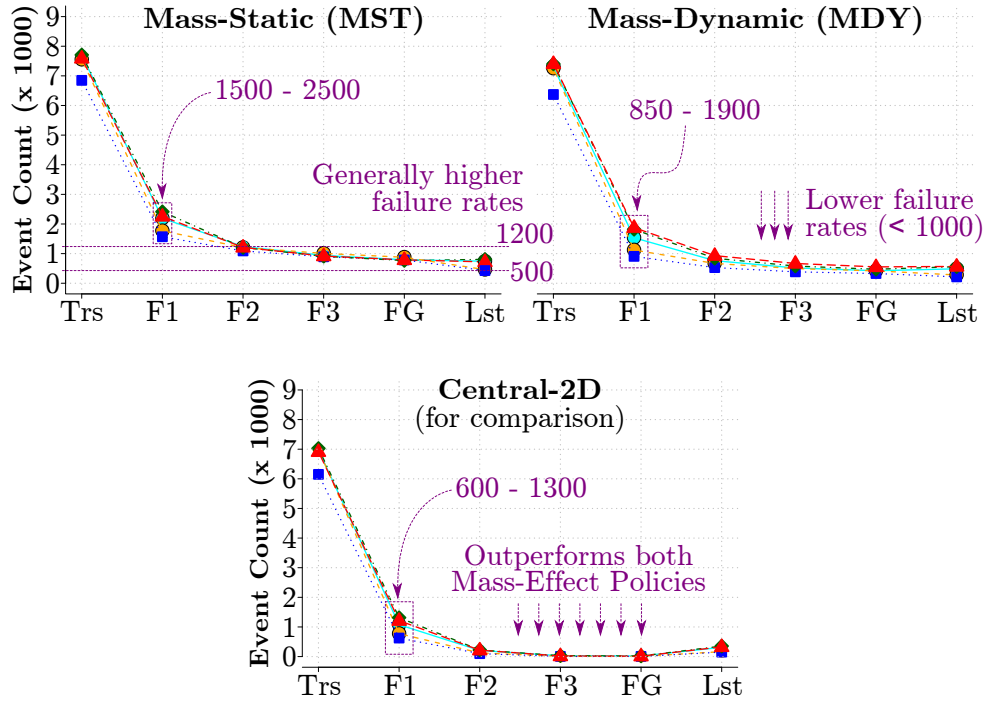


Figure 3.5: Failure rates for Mass-effect selection policies (static on left and dynamic on right) against C2D.

the arbitration request count. This is a mechanic of the simulation and is not an attempt on our part to artificially increase the number of tries. Nodes simply initiate more combat as prior attempts are rejected, reflecting a need to participate more intensely.

In Fig. 3.6, we see that Randomized selection (RAN) outperforms most of the prior schemes, with its plots very closely tracking that of C2D. This is perhaps due to its uniform nature, as there is a greater probability of selecting fit arbitrators from an *indiscriminate* pool of NBs. It is interesting to note how the simpler RAN scheme yields very similar results to the best-performing distance-based scheme (C2D), with much less computational complexity.

Again in Fig. 3.6, we immediately notice the very minimal failures incurred for the the Fittest (FIT) policy. The overall reductions are very pronounced, with later fail-events (Fail-2 to Lost) at or approaching zero instances and significantly reduced Fail-1 event

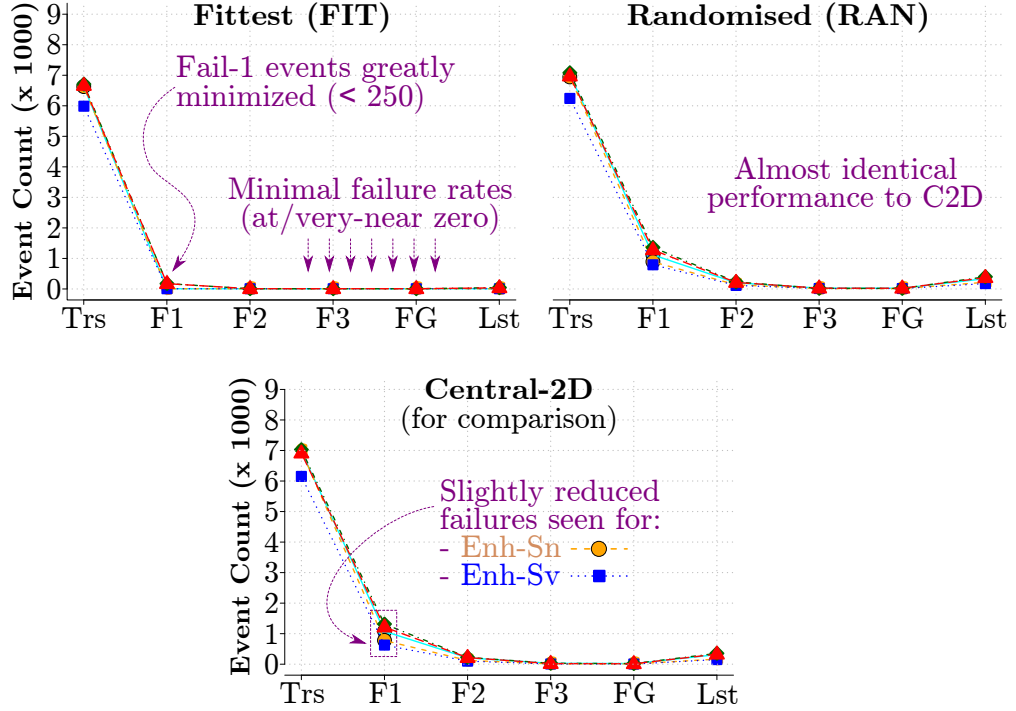


Figure 3.6: Performance of “Fittest” and the baseline random (RAN) policies.

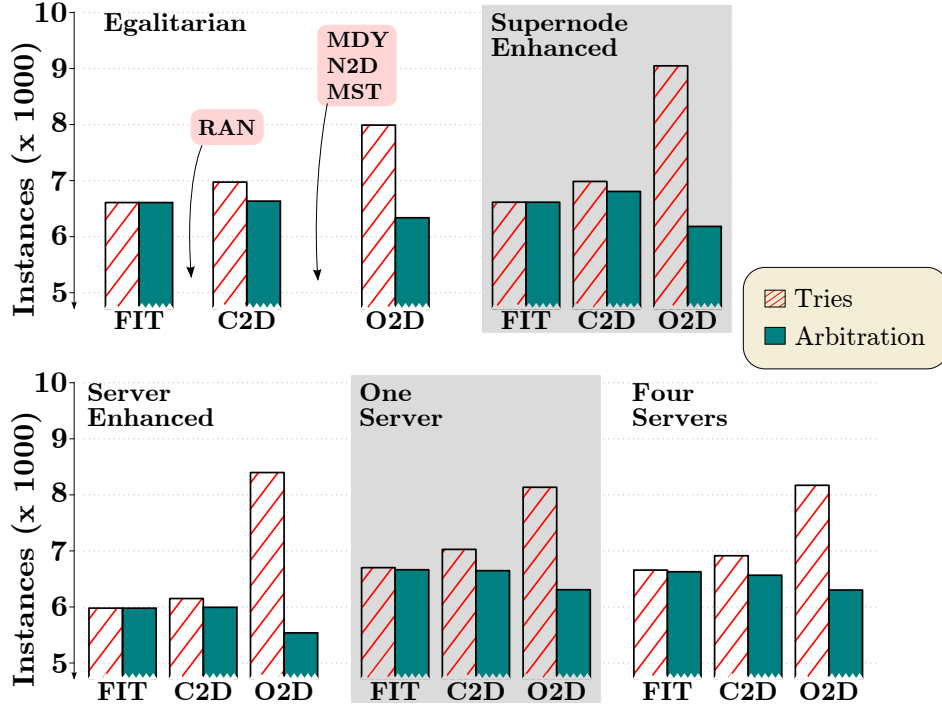


Figure 3.7: A comparison of *Tries* (on left of each *column pair*, red stripes) against successful arbitration (on right, solid green). Top row shows the server-less peer configurations, whilst the bottom row shows the general performance of server-enabled configurations.

counts. This reflects the ideal situation where NBs with the most available resources (i.e., topmost along the *Z*-axis) were habitually chosen *at the first attempt* and thus, minimizing overall rejections.

Nonetheless, failures were recorded due to specific instances where a multitude of peers requested arbitration from a particular peer, all within the same tick. Ergo, the unfortunate candidate at the top of the *Z*-axis is flooded with requests and is forced to reject some. This does represent a weakness in the FIT policy, as they will be times when high-capacity peers are suddenly overwhelmed. By always choosing the highest node on the *Z*-axis, we unwittingly enable this phenomena. However, on the whole, FIT is the best performing policy with the lowest failure rates. FIT minimizes retries and thus, any delays to the start of arbitration, thereby enhances overall VE responsiveness.

To understand the effects of inefficient policies, we refer to Fig. 3.7, where we see the difference in requests against actual fulfilment. Across all configurations, we see that FIT deals with requests easily. This culminates in lower total *Tries*, as peers have running interactions and do not initiate newer ones. This is vindication of our earlier argument for the desirability of servicing arbitration demands quickly and avoid the detrimental effects of repeated requests.

For clarity, only three marker policies were shown in Fig. 3.7, to highlight differences across the spectrum. For each simulation configuration, the order of performance is as follows: FIT (shown), RAN, C2D (shown), MDY, N2D, MST and O2D (shown). Generally, each design has its merits relating to responsiveness and/or security but high failure rates preclude the use of some.

C2D/3D sought the middle ground between responsiveness and security, balancing faster set-up times with the need for peers uninterested with local game-play. For the most part, they do present better results, minimizing failures of type Fail-2/3/G. Interestingly, blindly incorporating Z-axis values into distance-based schemes (N3D, C3D, O2D) proved inconsequential at best.

The mass-effect policies incorporated the dynamic capacity metric and were an ideal fit for our 3D domain. However, the strong denominator in the inverse square law (D^2) meant capable peers were continuously rejected. The simulated “gravitational” pulls were not enough to overcome great distances. This can be seen with the close tracking of results for MST with N2D/3D.

FIT proved the best performer but its success is tinged with the aforementioned issues with sudden request volumes. Additionally, the crucial need to intelligently select nearer or further arbitrators is left to pure chance.

The almost real-time responsiveness needed to maintain interactivity in online-games is what separates P2P-VEs from other P2P research fields. While a file-sharing system can

be tolerant of high latencies, online game-play is highly dependent on the timely delivery of updates [Steed and Oliveira, 2010]. It is eminently frustrating for players to press a button and only see their action registered a few seconds later. In such situations, all interactivity and immersion is destroyed. FPS games are particularly vulnerable in this regard, with [Armitage, 2003, Knutsson et al., 2004] noting that latencies beyond 150-180ms are detrimental to in-game performance.

Elaborating further, [Chen et al., 2009, Claypool and Claypool, 2006] state that different genres of games have varying tolerances to network latency. The RPG class of games are noted to have players more tolerant of such high latencies. This is due to inherent game mechanics, where accuracy and timeliness are *not* defining factors in interactions. In contrast, FPS players are noted for their acute sensitivity to latencies. Thus, it is postulated that if we can satisfy FPS players, then we can satisfy all others. This is the reason for our enduring focus on FPS-styled games.

As with current commercial games, the prevailing industry practice elevates responsiveness over game-play security, with basic client-server (CS) architectures the most widely employed amongst them. As articulated in [Sweeney, 1999, Bernier, 2001, Skibinsky, 2005], CS techniques simplify the issues of responsiveness and to a limited extent, security. This however, is at the expense of scalability and flexibility. Hybridised CS and P2P [Rooney et al., 2004, Skibinsky, 2005, Buyukkaya et al., 2009] solutions exists but are often less fluid and fraught with handover issues as peers live in semi-enclosed subsets of the virtual world.

Actual game-play security is not a primary concern of this paper. However, the specific issue of game-state fidelity (i.e., consistency and cheat-resistance) forms a fundamental requirement [McCoy et al., 2005]. The reason we advocate strongly for singular arbitrators is given by examining past approaches:

- The basic Lockstep protocol of [Baughman et al., 2007] may be used but it is susceptible to high-latencies in its participants. Within a turn, a peer may (for whatever

reason) be slow in committing its decision. Thus, the resolution process is delayed and responsiveness for other peers is negatively impacted.

- Various distributed and group-voting mechanisms have been mooted [Corman et al., 2006, Webb et al., 2008, Goodman and Verbrugge, 2008, Liu and Lo, 2008]. However, maximising responsiveness in the P2P-VE are often secondary objectives, if at all. Further, group mechanisms incur extra processing and communication overheads when using multiple arbitrators.
- Most prior art striving for fully P2P NVEs have focused more on addressing network scalability issues; with map-partitioning, interest management and load-balancing the primary concerns [Yu and Vuong, 2005, Buyukkaya et al., 2009, Legtchenko et al., 2010, Hu and Chen, 2011, Bharambe et al., 2008, Hu et al., 2006].

A summary of related work is presented in Table 3.3, where it is seen that there is a disconnect in research areas. Most work in the P2P-VE area targets network scalability issues, whereas efforts on the issue of game-play arbitration generally consider focus on hybridised architectures. [Corman et al., 2006, Baughman et al., 2007] do have a focus on pure P2P environments but favour the multiple arbitrator approach. The proposed approach here is an attempt to bridge the two research sectors.

The idea of multiple arbitrators, is to self-audit peers and thus, diffuse the effects of cheating. Conceptually, this is aggregated consensus where more trust is placed in group decisions than on singular authority. The concern here lies with increased overheads and latency issues amongst referees. If one is delayed, the arbitration process, and thus game flow, would be impacted.

Alternative approaches include peer auditing schemes proposed for hybrid CS-P2P systems [Liu and Lo, 2008, Goodman and Verbrugge, 2008]. Again here, the localised structures are intransigent and monitoring overheads are incurred. [Webb et al., 2008] uses

Table 3.3: Summary of related architectures and approaches

Name	Type	Arbitration	Remarks
Basic [Sweeney, 1999, Bernier, 2001]	CS	Singular Fixed Servers	Inflexible, failure sensitive, restricted maximum players, easy & mature technology
Federated-P2P [Rooney et al., 2004, Skibinsky, 2005]	Hybrid	Singular Zonal Servers	CS with limited P2P in partitioned zones, restricted VE cohesion, zonal handover issues
IRS [Goodman and Verbrugge, 2008], Da-CAP [Liu and Lo, 2008]		Peer Auditing (Multiple)	Monitoring overheads, elements with CS structures, unnecessary inter-arbitrator latencies
SRS [Webb et al., 2008]		Selected (Multiple)	Inter-arbitrator latencies, dynamic selections but from a central authority
SEA [Corman et al., 2006], LockStep [Baughman et al., 2007]	P2P	Synced (Multiple)	Fully distributed, focused on fairness and security, only as fast as its slowest arbitrator
MOPAR [Yu and Vuong, 2005], Vorogame [Buyukkaya et al., 2009]		Not Addressed	Interest-management for node discovery & efficient updates; Uses Distributed Hash Tables (DHT); 2D-VD map partitioning in latter case.
Blue Banana [Legtchenko et al., 2010]			Attempts to improve VE responsiveness via player movement prediction mechanism.
VSO [Hu and Chen, 2011]			Uses 2D-VD for spatial publish-subscribe (update dissemination); Tackles load & scalability issues.
Donny-brook [Bharambe et al., 2008]			Interest-filtering to manage scalability issues; Updates to 900+ peers for large-scale combat.
VON [Hu et al., 2006]			Only addresses scalability & consistency issues; Single hop links assumed to be responsive.

inter-peer round-trip times as the main criteria for selecting “referees”. Crucially, they rely on a global authenticator (making it only partially P2P) and also use multiple arbitrators.

With most recent work enabling P2P principles on NVEs, the focus has been to assuage exponential bandwidth growth as potentially thousands of peers become inter-connected. This is arguably the cardinal concern, evidenced by the large volume of prior art and a very active research community.

A common thread that runs through most recent work is the concept of interest management, all in an effort to reduce the amount of bandwidth/resources used. The idea is to limit the scope of individual peers, to an extent where information dissemination and control becomes a more scalable proposition.

Earlier work in [Yu and Vuong, 2005] (MOPAR) merges unstructured network overlays with Distributed Hash-Table (DHT) principles. VoroGame [Buyukkaya et al., 2009] proposes a similar strategy, with the exception that map partitioning is done with 2D-VD. Both introduce the idea of a “controller” peer that is in charge of specific areas in the map, which may be construed as arbitrators in a sense. However, with their focus on scalability issues, little mention is made of the myriad of issues affecting the selection of these controllers and their suitability as arbitrators.

Donnybrook [Bharambe et al., 2008], is another well-known treatise in the area of P2P games. However, the focus there is again on interest management and the attempt to reduce bandwidth usage as the number of peers scale higher. While they successfully support the constant updating of a multitude of peers, little is said regarding the verifiability of the updates and the role of arbitrators.

Fundamentally, the work presented here is an augmentation of VON [Hu et al., 2006] by Hu *et al.* Besides the ability to incorporate the 3rd dimension, the fluidity of their network overlay is a key feature, as we prefer the more flexible clustering it affords. It does away with the rigid/semi-rigid map partitioning seen in previous works and its inherent

dynamism belies its simplicity. [Hu and Chen, 2011] represents a progression of their work to tackle load-balancing issues in P2P systems.

Chapter 4

Network Traffic Reduction

As prior chapters have indicated, the use of 3D-VD can cause an upsurge in bandwidth requirements, particularly as peers need to send update messages to a higher number of neighbours exposed by 3D-VD. This is compounded by the fact that by moving towards a *fully* P2P-VE, the amount of network traffic generated at each peer is a significant concern. Multiplayer Online Games (MOG) are the largest application subset of VEs and have been shown to require high frequency of update messages and minimal network latencies. Yet, this demanding criteria must be balanced with the need to also limit the otherwise quadratic growth of network traffic amongst peers.

Two-dimensional Voronoi Diagrams have been proposed as a way to address the inherent traffic scalability issues by naturally clustering players (and thus their update traffic) within the game-world. However, other important issues related to game-play and overall VE performance remained and were only addressed by our recent introduction of a third dimension to the VD computations (3D-VD). As our experimentation indicates, this unique approach has significant impact on the network characteristics of a P2P-VE. Due to its 3D nature, more connections are necessary per peer but a mechanism is successfully introduced to cope with the increased bandwidth requirement. More importantly, the results obtained show considerable reduction in traffic load under varying peer topologies while still maintaining the desirable features of 3D-VD.

4.1 Impact of The Z-axis

The third dimension, commonly referred to as the Z -axis is the main differentiating factor of our approach. Prior art has only utilised the corresponding X and Y axes to represent a peer's in-game location. Our use of the Z -axis is not to simply add the in-game Z data into the VD derivations. Instead, a real-world, non-game related and independent metric is used. However, the mere introduction of a third axis has implications on the computational complexity of any algorithm and the resulting solution representations.

It is noted that 2D representations may be computed in a timely manner, up to the order of thousands of peers [Shewchuk, 1999, Hu et al., 2006]. However, as is the case with most computational geometry problems, deriving higher order VDs becomes progressively harder and computationally intensive with each additional dimension added. Every single peer within the VE needs an accurate VD representation of the space around it and lengthy computation will negatively impact game-play.

The scale of the Z -axis also plays a significant role in determining the resultant DT. For example, a DT derived when the Z -axis is within the unit range of 0 to 100 is quite different to one obtained when the Z values are from 0 to 1000. This is due to the inherent properties of a DT where the angles forming a triangle (or tetrahedrons in the case of 3D) tend to be as obtuse as possible. Thus, when the Z -axis is “stretched”, the obtained tetrahedrons are different and as a consequence, so are the subsequent VDs. We address this issue by normalizing the third dimension between predefined minimum and maximum values across all peers, resulting in a fixed and standardized Z -axis.

4.1.1 Metrics

The question then becomes: What metric is to be used as the third dimension? Any candidate must be *independent*, both in its relationship with the X and Y axes and also in the relationship *between peers*. That is, the X and Y values and any direct relationship

measure between peers should not immediately influence the computation of the Z values.

For example, using a formula such as $z = \sqrt{x + y}$ would cause the peers to artificially clump together and degrade the usability of the resultant 3D-VD. In this case, computation of the Z axis is entirely dependent on the other two axes. In another negative example, if we were to use the ping time from one peer to another, the metric would then become a *relative one* amongst peers. As the round trip time (RTT) is, by nature, a volatile measurement between peers, using it would cause unnecessary fluctuations on the Z -axis and the 3D-VD.

As we are primarily concerned with the network traffic issue in this paper, we chose to run our simulations using bandwidth capacity as the third dimension. This metric is a measure of a peer's *remaining upload capacity* at any given moment in time. To further explain, should a particular peer have no direct neighbours to connect (and thus, send updates to), it would then be higher on the Z -axis (up to its maximum value). When it needs to connect to more and more fellow peers, its Z value progressively diminishes to a minimum of zero (indicating that it has reached its upload limit).

This metric is divorced from the positional values used on the X and Y axes and instead is a by-product of the dynamic connection requirements of the peer throughout its life within the VE. There are, of course, other independent measures that may be utilised. The reader may refer to earlier sections for more discussion on possible metrics.

4.1.2 Peer vs Global Maxima

It must be noted that there is a distinction between the maximum theoretical values for particular types of peers and the overall maxima used for the Z -axis. A peer's maximum network upload capacity only relates to itself and is a direct function of its type (i.e., if it is a normal node; a weak, constrained node; or an actual dedicated game-play server).

Thus, a constrained peer can never reach as high as typical peers (normal nodes) on the Z -axis. Likewise, normal nodes can never reach the levels of super-nodes and they, in turn, can never reach the Z values of dedicated game-play servers.

Table 4.1: Comparison of simulation topologies

Category	Description	Servers	Peer Count	Super Nodes	Normal Nodes	Weak Nodes
Egalitarian	All Constrained Nodes	0%	1000	0%	0%	100%
	Evenly Mixed	0%	1000	0%	50%	50%
	All Normal Nodes	0%	1000	0%	100%	0%
Enhanced	With Supernodes	0%	1000	40%	30%	30%
	With Servers	10%	900	20%	50%	20%
Minimal Servers	Single with All-Constrained	1	999	0%	0%	100%
	Single with Egalitarian	1	999	0%	100%	0%
	Four with All-Constrained	4	996	0%	0%	100%
	Four with Egalitarian	4	996	0%	100%	0%

However, movement downwards to the minimum Z value is entirely possible. A server (and every other type of peer) can eventually reach the lower levels of the Z -axis and become on par with weak-nodes. This happens when a server becomes ever more loaded due to an increasing demand for its arbitration services (e.g., when normal peers select it to adjudicate interactions between them), thus resulting in it slipping further down the Z -axis. Becoming an arbitrator demands more bandwidth resources as peers send updates to the servers which, in turn, have to process their modified states and propagate these decisions to all affected peers.

Figures 4.1a and 4.1b aptly illustrates this phenomena (nodes seen in these figures were generated for illustrative purposes only). Note the overlaps between the different peer types and it is shown how some servers have almost reached the lower regions of the Z -axis. Figure 4.2c is representative of a non-enhanced topology where there are no servers and super-nodes present.

For the purposes of our experimentation, the global maxima is taken from the maximum

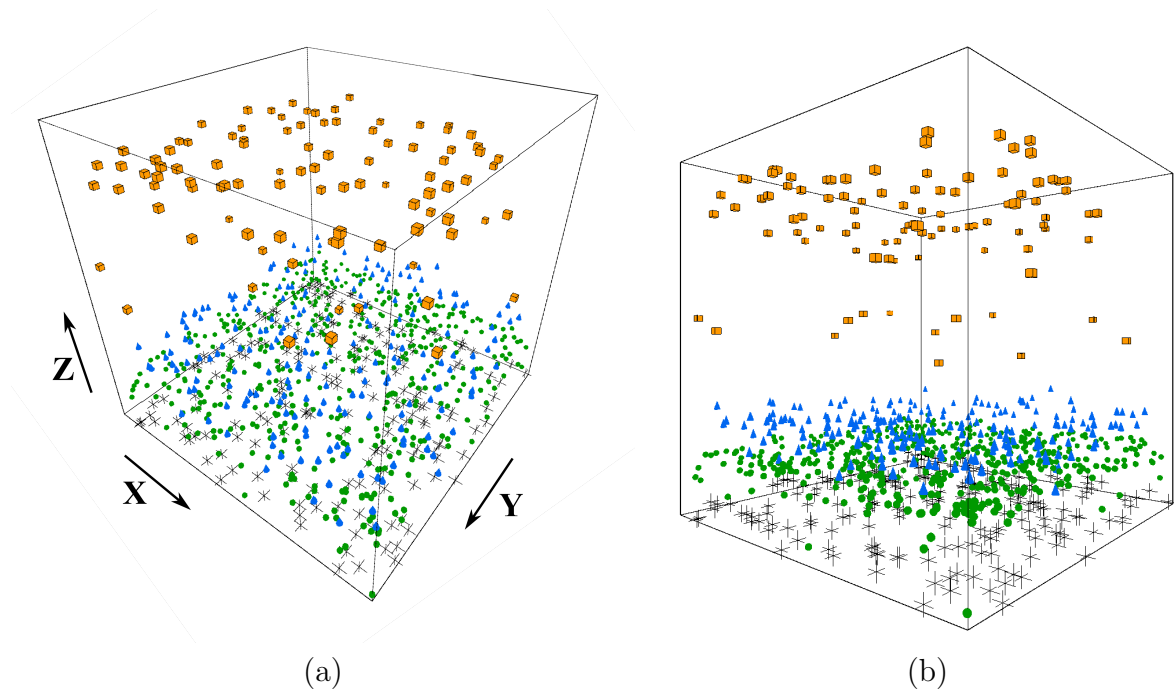


Figure 4.1: A 3D View of a mixed, server-enhanced peer topology consisting of 10% servers (orange cubes at top), 20% Super-Nodes (blue cones), 50% Normal Nodes (green dots) and 20% Weak Nodes (crosses) is shown in (a). (b) further exposes the Z-axis to better illustrate the distribution of peer types according to current bandwidth capacity.

value possible on servers. All other peer types have their Z values normalized against this global value. The following maxima were used for each peer type:

- **Weak Nodes:** 50% of the capacity of a Normal Node
- **Normal Nodes:** reference value (100%)
- **Super Nodes:** 150% of normal capacity
- **Servers:** 400% of typical normal-node capacity

Thus, should we assign the value of 1 Mbps to a typical (i.e, normal) node, the servers would have a 4 Mbps upload capacity. Further varying the actual capacities of individual

peers and studying its effects is the focus of future work. Indeed, it would make for interesting study but for simplicity and brevity, only these simplified pre-defined ratios are used in our full experimentation. Our obtained results indicate no noticeable disparity between simulation-runs where we varied the proportions.

4.1.3 Network Topologies

There is a need to test a 3D-VD enabled system across a range of network topologies consisting of differing distributions of peer types. This is in an effort to characterise the make-up of real-world peers in any potential P2P-VE. More importantly, we wish to also expose the operational behaviour and the impact of using our modified third dimension. The various configurations implemented in our experimentation can be broadly categorised into:

- **Egalitarian:** Non-enhanced configurations where nodes have either normal or con-

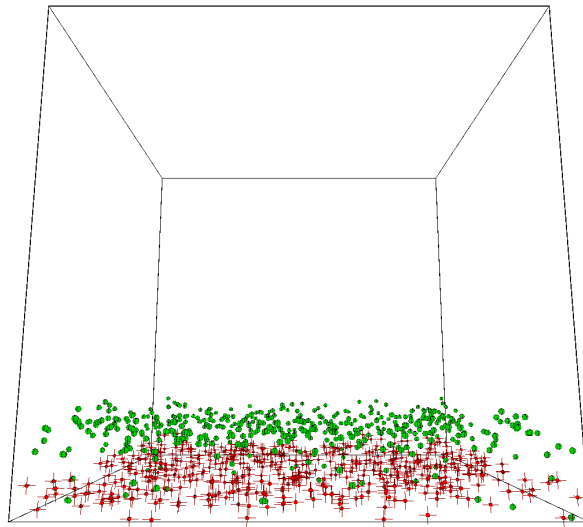


Figure 4.2: Shows a mixed-egalitarian topology where only normal-nodes and weak-nodes exist in a 50-50% distribution. Note the absence of any high-capacity peers inhabiting the top of the cube.

strained capacities.

- **Enhanced:** where peer populations are seeded with either super-nodes or dedicated servers.
- **Minimal Servers:** where very minimal numbers of dedicated servers are inserted into the network.

Table 4.1 lists the various sub-topologies involved and details the peer count against that of servers and the distribution percentage of the remaining unallocated peers. It must be noted that the servers remain static within the virtual world (they do not move along the X and Y axes). This is due to the fact that they are meant to service game-play amongst participating peers. As a result, there is no reason for them to move about in a similar fashion to the other peers. Alternatively, this behaviour may be easily modelled and is, in fact, actively investigated in ongoing parallel work.

For the “minimal-servers” configurations, sole servers were intentionally positioned in the middle of the virtual world. That is, their positions remain static at $S_x = 1000$ and $S_y = 1000$ in a virtual-world map with XY dimensions of 2000 by 2000. For topologies with four active servers, the map is split into quadrants and the servers placed in the middle of each (i.e., $S1[x, y] = \{500, 500\}$, $S2[x, y] = \{1500, 1500\}$, etc). In the server-enhanced topology where 10% of the peer population are converted to servers, they are placed randomly and again remain static throughout the lifetime of the simulation.

While the number of configurations is limited, it is hoped that they are indicative of peer distributions in any conceivable P2P-VE. The main thrust of our work is to preclude the use dedicated game-play servers and the obtained results (detailed later) support our approach. The need for comparison is the main reasoning behind including server configurations.

Table 4.2: Expected Number of Neighbours

Topology Type	Average	Min	Max
Egalitarian (mixed)	73.57840	48.8	110.28
Enhanced (supernode)	77.40005	51.4	158.55
Min-Servers (One & Constrained)	67.79891	42.65	93.86

4.2 The issue of increased bandwidth

The most salient impact of adding a third dimension to the VD computations is the increased number of neighbours. Due to its nature, a 3D-VD exhibits a tendency to force peers into maintaining a larger number of direct connections with its neighbouring peers. As outlined by Hu *et al.* in [Hu et al., 2006], all peers within the VE *must* retain communication links with both its *enclosing and boundary* neighbours. This would preserve game-play consistency and prevent peers losing connections and forming islands that are not ultimately connected to each other in the virtual world (a process that would lead to irrecoverable fragmentation and VE failure).

Table 4.2 lists the average number of neighbours (both enclosing and boundary) that a peer is expected to connect to. The numbers were obtained in our simulation runs and is the mean value across all 1000 peers throughout their lives in the network. Of particular concern is the high maximum values recorded indicating that there are instances where peers were forced to simultaneously send updates to 150 peers.

As reported in [Bharambe et al., 2008], update messages are delta-encoded IP packets totalling 80 bytes per update. Multiplying by the oft-quoted industry standard of 30 frames per second (fps), this equates to an upload requirement of 360 KBps or 2.88 Mbps. This upper bound is well beyond the typical 1 Mbps upload capacities of current home broadband services.

4.2.1 Neighbour Classification

Thus, the need is there to effectively reduce the traffic generated per peer. The first step of our approach is the classification of the neighbours (NB) that a particular peer must connect to. Figure 4.3 illustrates the difference between the enclosing and boundary NBs of a peer. Enclosing neighbours (Enc-NB) are fellow peers in the virtual world that directly envelope the peer computing its 3D-VD (P_{vd}). That is, these peers are first-tier NBs and are the closest in proximity to P_{vd} .

Boundary neighbours (Bou-NB), form the second-tier in the classification. They fully envelope P_{vd} in the 3D space as well and form a larger convex hull. However, the 3D-VD representation indicates that there is an intermediary NB between them and P_{vd} . This is how we differentiate between NBs and, as detailed later, the way we can significantly reduce the communication requirements of the target peer, P_{vd} . Any other peer outside the convex hull formed by the Bou-NBs are *not* considered NBs and thus, P_{vd} is not required to maintain communications with them.

It must be noted that there will be situations where NBs deemed to be boundary, are actually right next to P_{vd} in the virtual world. For example, if we were to flatten Figure 4.3c along the Z -axis, several Bou-NBs would in fact be right next to the central computing peer. A 2D-VD would immediately classify these NBs as *enclosing* whereas the 3D-VD approach would dismiss them as Bou-NBs. In practice, it is expected that such occurrences would cause a 10 to 15% shift in the classification from Bou-NB to Enc-NB.

4.2.2 Varying Upload Rates

We have thus outlined a deterministic method to categorise a peer's neighbours into two sub-types. The task now is to *vary the rate* with which we send update messages to them. The approach is relatively straight-forward in that we simply treat Enc-NBs with a much higher priority than Bou-NBs. The assumption is that Bou-NBs have an intermediary

layer, comprising of other active peers, between them and the central computing peer (i.e., P_{vd}). Thus, they are likely to be of less interest to the human player at P_{vd} .

On the other hand, the Enc-NBs are the closest in proximity and thus, likely to be of higher interest to the player. For example, in a setting involving the use of only melee weapons, a player would only be concerned with hitting the opponents that are closest to him. Thus, movement and actions of his direct opponents should be prioritised. Likewise, the opponent uses his 3D-VD to prioritise *his* Enc-NBs (which would include the player) and send *them* updates at a higher rate.

Algorithm 1: Prioritised Rate of Updates

```

 $P_{vd} \Leftarrow$  Computing Peer
 $t_{max} \Leftarrow$  1 second
while  $P_{vd}$  is active do
  Receive updates from current NBs
  Derive  $VD$  from positional data
  for all  $NB_i$  in  $VD$  do
    if  $NB_i$  is ENCLOSING then
      Send Update to  $NB_i$ 
    else if  $NB_i$  is BOUNDARY then
       $t_i \Leftarrow$  Get Last Update Time
      if  $t_i > t_{max}$  then
        Send Update to  $NB_i$ 
      else
        Skip Sending Update
      end if
    else
      Drop from NB List
    end if
  end for
end while

```

It can be inferred from Figure 4.3 that there will be a larger proportion of Bou-NBs. This property holds true across all configurations and thus, allows us to effectively utilise our upload variance algorithm, as outlined in Algorithm 1. The upload bandwidth required per second can thus be shown as:

$$BW_{up} = (|NB_{enc}| \cdot P_\delta \cdot F_\epsilon) + (|NB_{bou}| \cdot P_\delta \cdot F_\beta) \quad (4.1)$$

where,

P_δ Packet Size (delta-encoded)

F_ϵ Update Frequency for Enc-NB (per sec)

F_β Update Frequency for Bou-NB

We are thus relying on the fact that there will always be a larger ratio of Bou-NBs. We then use this property in a manner which reduces the upload traffic yet maintains the connectivity of the peers in the game-world. For simplicity, our simulations were run with $F_\epsilon = 30$, to keep in line with the 33.33 milliseconds limit (per graphic frame rendering) that modern games often strive for. That is to say, positional updates between Enc-NBs must be sent every 33 msecs in order to maintain optimal responsiveness in game-play.

For the frequency of updates to Bou-NBs, F_β is set simply to the value on 1. Meaning updates to Bou-NBs are sent at a much slower rate of only once per second. In this regard, we follow the methodology in [Bharambe et al., 2008] where non-primary peers are treated

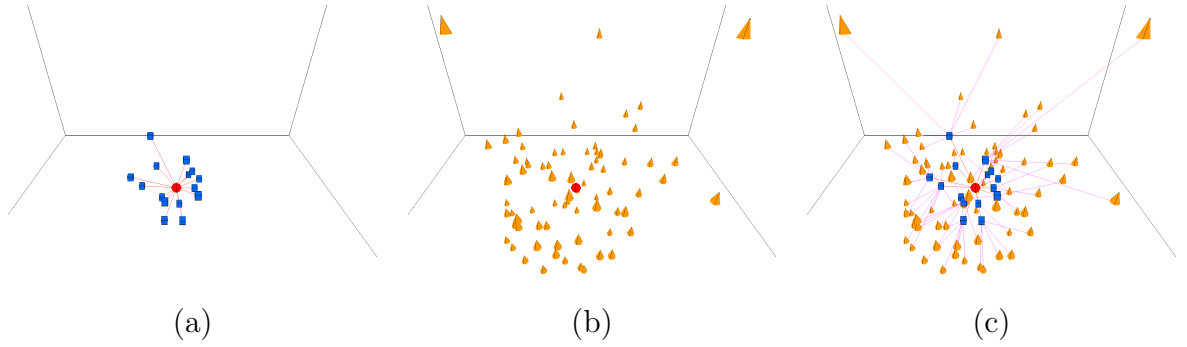


Figure 4.3: Showing the dual-tiered classification of the neighbours (NB) of a peer and their relative numbers. The peer computing its 3D-VD is the central red sphere. The blue cubes in (a) represent the *enclosing* NBs whereas the yellow cones in (b) are the *boundary* NBs. (c) shows the fully interconnected graph (note: nodes are illustrative).

in a similar fashion. Although they utilise a prioritised rate of 20 updates per second (ups) for primary peers, their secondary tier of peers are also sent updates at 1 ups.

4.3 Experimental Results

The reader would by now realise that we are, in effect, implementing a unique variation of interest management/filtering techniques. As done in [Bharambe et al., 2008, Keller and Simon, 2002, Hu et al., 2006], the aim is to cluster and classify peers and vary the amount of data transfer based on the results. The compelling argument with our approach is that we are able to augment the use of Voronoi diagrams yet retain the elegant filtering techniques afforded by it. The use of the Z -axis gives an added angle with which to tackle the other P2P-VE issues and thus, is a desirable characteristic.

Other approaches use a variety of non-deterministic factors, are somewhat predictive in nature and sometimes disregard vital aspects of game-play. Using 3D-VD is a simple, computationally feasible and strictly deterministic method of interest filtering which uses peer proximity as the main distinguishing criteria. There is no need to “guess” player interaction levels, expected targets and the like. Peers are naturally clustered according to their position in the virtual world and basic communication requirements are based from that. Another advantage over the implementation by [Bharambe et al., 2008] is that we do not need to send updates to *each and every other peer* in the virtual world. The sending of updates is inherently localised and there is no need to incur additional bandwidth costs sending updates to peers on the other side of the map.

4.3.1 Test Set-Up

Preliminary testing as reported earlier indicated that it is feasible to compute and use 3D-VDs within the (almost) real-time constraints of a DVE. The initial worry was that if not

done in a timely manner, 3D-VD derivation will hinder the responsiveness of the game-play by increasing the processing delay at each individual peer.

This concern was assuaged in earlier chapters, where we approached the issue by developing a software prototype which leverages on the open-sourced Computational Geometry Algorithms Library (CGAL) [Pion and Teillaud, 2009]. It was shown that obtaining the 3D-VD within the acknowledged upper-bounds of 33 msec was entirely possible. Our experimentation here was built on that software prototype and the testbed (as then) was a Linux desktop running an Intel Core 2 Duo processor (four cores @ 3000Hz) with 4GB of RAM. Due to the nature of CGAL, C++ was the development language with the Qt C++ API providing most of the application back-bone.

We were obviously not able to run real tests with 1000 peers in a network. Simulations were instead run for 1000 peers across the total of nine network topologies over a period of 500 consecutive “ticks”. Each tick here can be construed as a slice of “simulation time” where the actual real-world time could be adjusted as required. Throughout our experimentation, we take each tick to correspond to 33.33 milliseconds in order to reflect the standard minimum target of 30 fps for present-day FPS games. This allows it to fall in line with the stated pre-defined F_ϵ of 30 ups.

4.3.2 General Trends

Figures 4.4, 4.5 and 4.6 shows the probability mass functions of recorded uploads per peer, per second. We can see all topologies have plots following a normal distribution, with typical upload bandwidth at around 1.5 Mbps. In contrast, the modified bandwidths reported in Figure 4.7 show PMFs with steeper peaks at around 0.3 Mbps. For clarity here, only three representative topologies are shown but the plots for other topologies follow the same trends.

It can be seen from the PMFs that the recorded upload bandwidths generally agree with the features of a normal distribution. While we hesitate to declare this as evidence

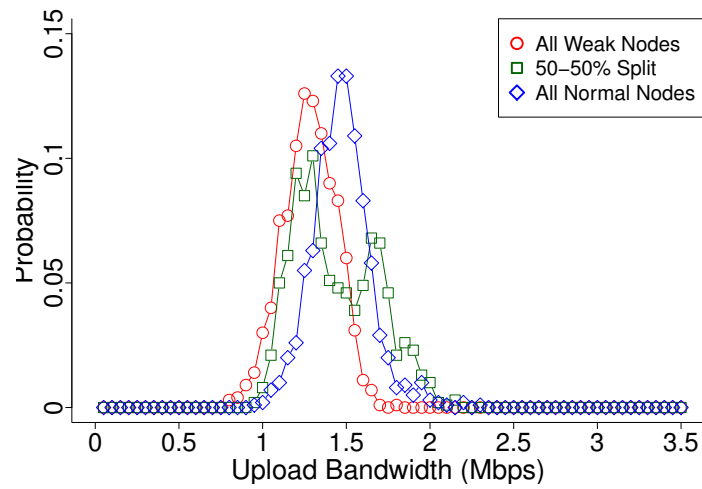


Figure 4.4: Probability Mass Functions of recorded upload bandwidths for completely server-less topologies

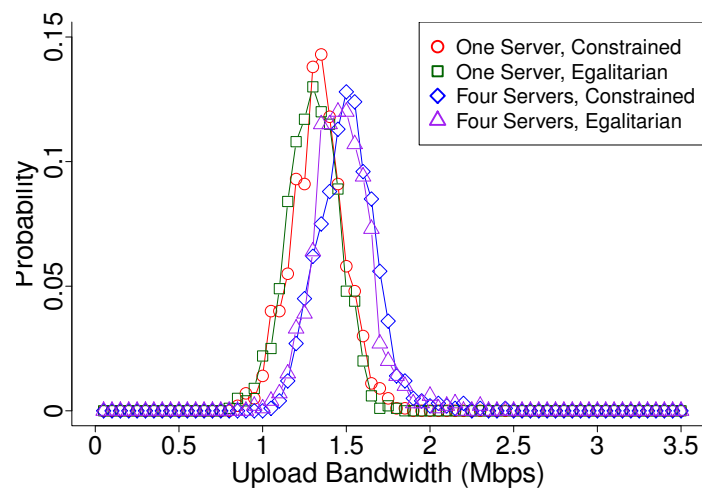


Figure 4.5: PMFs of configurations with minimal servers present (1 and 4)

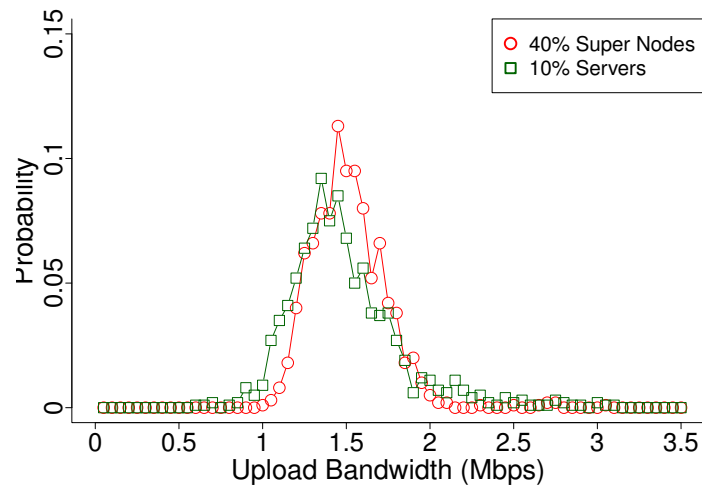


Figure 4.6: PMFs of configurations with 40% Supernodes (Super-node Enhanced) and 10% Servers (Server-Enhanced)

of the correctness of our experimentation, it does validate our system and our general

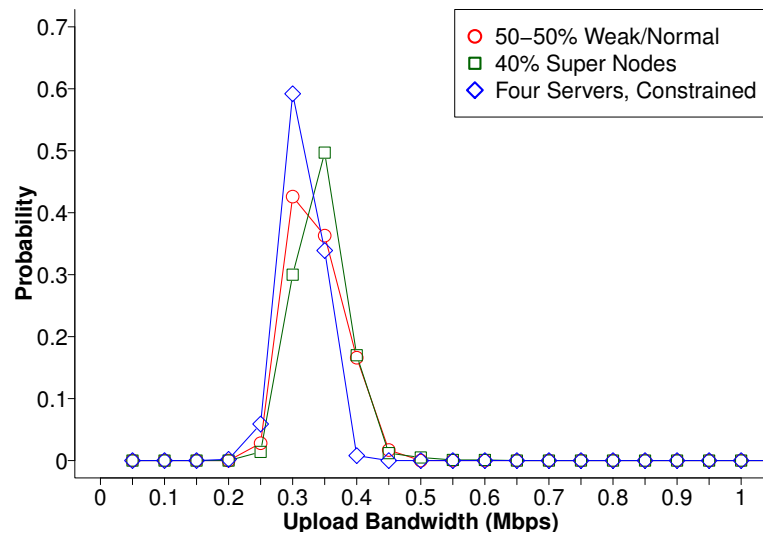


Figure 4.7: Probability mass functions of three representative topologies running the enhanced updating algorithm. Note the general mean BW values as opposed to the ones just earlier (0.3 to 0.35 compared with 1.5 Mbps)

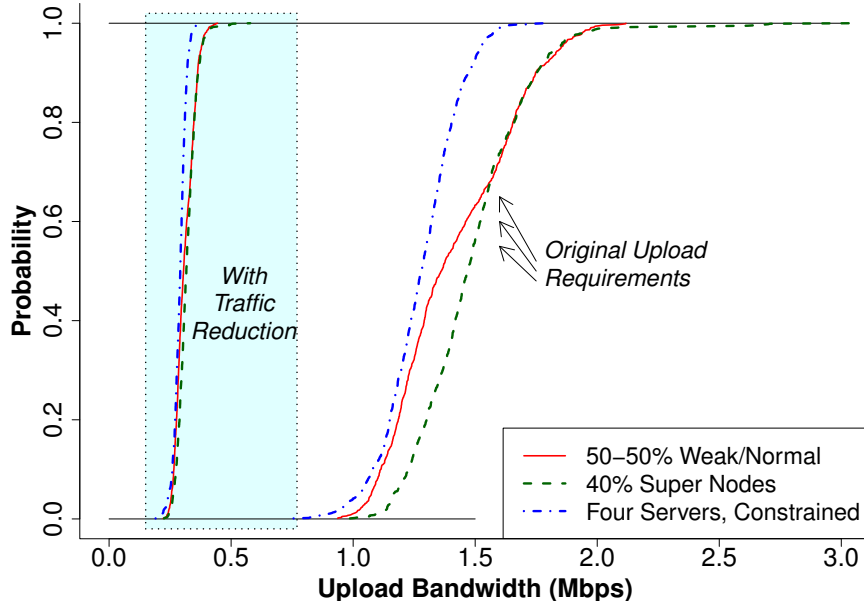


Figure 4.8: Cumulative Distribution Functions of three representative topologies, where the left most plots display the reduced bandwidth requirements and the plots on the right are their original recorded values.

methodology to a certain extent. Of note is the generally smoother gradients and smaller peaks seen in the enhanced configurations seeded with 40% super-nodes and 10% servers (Figure 4.6). This owes to the fact that these topologies have a better spread of peers and servers along the Z -axis (refer to Figure 4.1). Thus a situation is produced where servers at the higher end of the Z -axis tend to *stay* on the higher axis as other servers are utilised.

Another interesting result is the double peak seen in the plot for the unmodified 50-50% split between normal and weak nodes (Figure 4.4). One possible explanation is that a number of normal-nodes are being continuously pulled down into the zone comprising of weak-nodes, as they demand arbitration services. As there does not exist more powerful peer-types, these normal-nodes are forced to fulfill the arbitrator duties. This incurs more bandwidth costs in the select normal-nodes and thus produces the second peak seen at 1.75 Mbps.

A summary of the general trends can be seen in Figure 4.8. It displays the empirical cumulative distribution functions of the recorded average upload requirements per peer. Three representative topologies are shown with a view to compare their trends with and without the use of our reduction algorithm. The plots on the left are obtained from simulation runs with the modified upload rates. It can be seen that there are dramatic reductions when using peer classification mechanisms.

It is interesting to note with the plots on the right in Figure 4.8, that there seem to be a slight reduction when using 4 servers as opposed to other configurations. It must be noted that the configuration seen is that of four servers and 996 *weak* nodes. The nodes would tend to accumulate at the bottom of the Z -axis and thus maintain a similar number of neighbours as the move around in the virtual-world. The four servers, on the other hand, do most of the heavy-lifting with regards to arbitration services. Thus, the general trend is for reduced bandwidth for the majority of peers, as evidenced in the plots. This would indicate that it might be better to seed the topology with some dedicated servers in order to achieve some bandwidth reduction. However, the reduction is only slight and the vast majority of peers still overshoot their theoretical upper limits.

4.3.3 Per Peer Reduction

Figures 4.9 and 4.10 offers a more detailed view of the circumstances of each peer and shows where most of our claimed reductions come from. It is seen that when using the naive approach of updating at a constant rate for all peers, each peer's bandwidth consumption go beyond the 1 Mbps limit (denoted by the horizontal dotted line). In fact, the average bandwidth used can be seen nearer to the value of 1.5 Mbps, with updates to Bou-NBs taking a lion's share of the overall utilisation. In Figure 4.10, we see the effect of slowing the rate of updates to Bou-NBs where consumption is reduced to well below the limit to values nearer to 0.25 Mbps. It is also noted that most of the variance in recorded values are due to the updates to Bou-NBs whereas Enc-NB bandwidth utilisation remains unchanged.

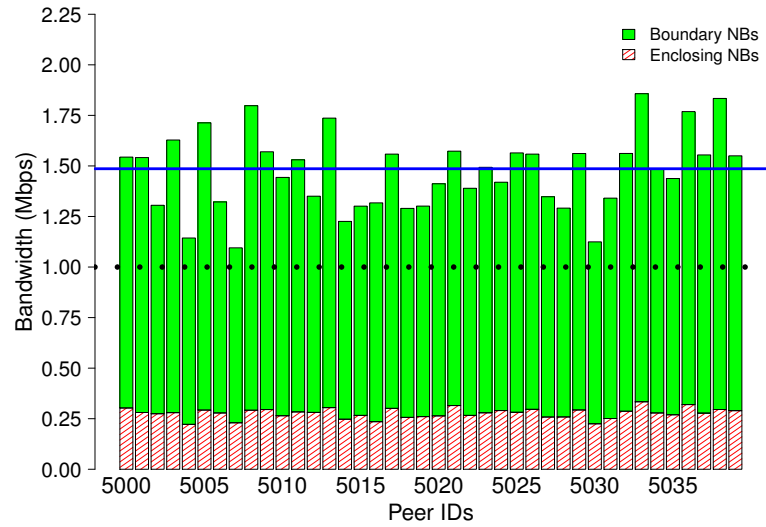


Figure 4.9: Illustrates the bandwidth (BW) reduction per Peer and per Tick. Here, we see boundary neighbours (NB) are sent updates at the same rate as enclosing NBs.

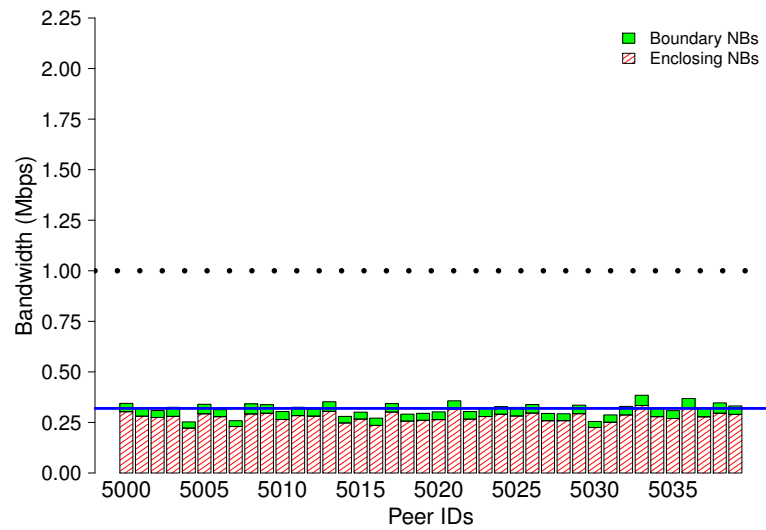


Figure 4.10: Illustrates the subsequent reductions shown with our approach, with the average BW per peer (solid horizontal line) dropping to well below the theoretical limit (dotted line).

For clarity, these bar plots show only a small cross section of the peer population, with peer IDs ranging from 5000 to 5040 (to a maximum of 5999). They are indicative of the rest of the peer population in that particular simulation configuration. Figure 4.10 shows a cross-section of peers in a configuration where 10% of the population are of the peer-type “server”. Note how Peer-5001 has a seemingly higher bandwidth consumption approaching 2.25 Mbps. Secondary checks reveal that Peer-5001 is indeed one of the many servers.

Investigating the raw data for that topology, we found that the servers in such topologies tend to have higher recorded values due to their positions in the 3D-VD. Being the only ones high up on the Z axis forces them to maintain connections with a higher number of fellow nodes. They are simply more exposed to the peers below them. In contrast, a typical normal-node would have a more limited scope by virtue of being in and amongst its fellow normal-nodes.

This is a phenomena of 3D-VD and leads us to conclude that minimal server configurations are to be avoided; the servers are simply being made update to more peers by the

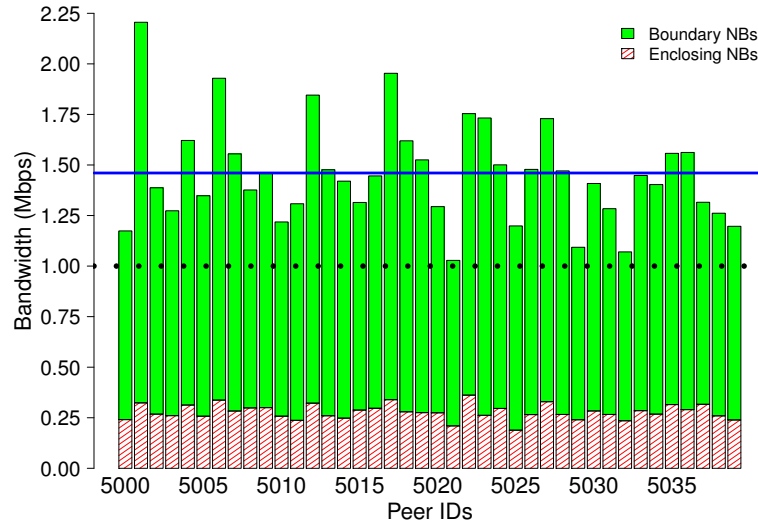


Figure 4.11: The peers shown here are taken from a different configuration, where 10% of peer population are servers (accompanying reduction plot is omitted).

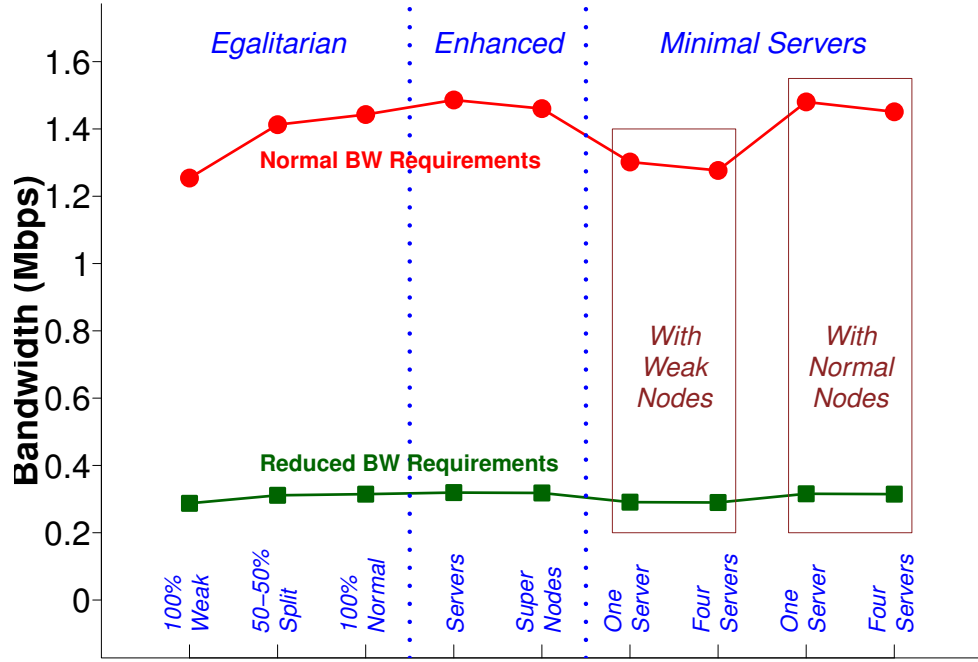


Figure 4.12: Reduction comparison across all topologies

3D-VD. For example, Peers 5006 and 5012 are also servers and thus, also record higher average uploads per tick.

Finally, Figure 4.12 plots the mean of the recorded values across all simulated topologies in a categorised manner. The reductions can be seen across the board and we note the higher mean values attached to configurations with normal-nodes. Topologies with higher percentage of such nodes tend to have higher mean upload requirements. An explanation could be that normal nodes have more room to fluctuate along the Z -axis, thus causing a greater spread and ultimately, more Bou-NBs to connect to per peer.

Also, as evidenced by the higher plot, we can see the inherent variability of bandwidth utilisation when Bou-NBs are treated equally as Enc-NBs. Observing the smoother lower plot, it is noted that our rate-prioritisation technique not only dramatically reduces the per-peer upload capacities required but also reduces their variance. This is no doubt a desirable

characteristic as engineering the tolerances required per-peer becomes much easier in any future P2P-VE implementation.

Chapter 5

Load Management

This chapter considers the questions posed by our fourth research question, that of improving the load management capabilities within P2P-VE. The focus here is to return to one of the fundamental goals of P2P research, the distribution and control of load across a heterogeneous peer population. With the application of 3D-VD, we can pursue this aim within the domain of on-line games and virtual environments.

5.1 Background

Distributed VEs (DVE) remain an exciting frontier of research, exposing many theoretical problems due to its unique constraints. Foremost is its time-sensitivity, resulting from the need to provide a responsive experience to users [McCoy et al., 2005]. The most stringent subset of DVEs is that of First-Person Shooter (FPS) games, where the game-play emphasizes fast reflexes and accuracy. This, in turn, leads to the need for minimal latency connections between participants and the server, as noted by [Knutsson et al., 2004].

This is in contrast to Massively Multiplayer Online Role-Playing Games (MMORPGS) such as the well-known World of Warcraft. Players of such games are more tolerant of network lag [Chen et al., 2009], due to game-play which de-emphasizes aiming accuracy. Such MOGs regularly capture the attention of the public and indeed, academic researchers.

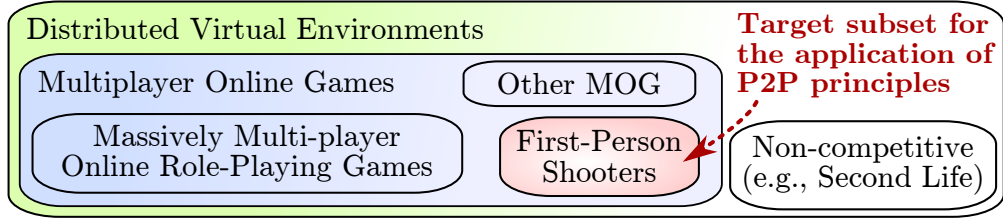


Figure 5.1: Overview of various subset application areas in DVE research.

For industry, DVEs are often quoted as a driver of growth in Internet services [Ross, 2009].

The default industry approach heavily incorporates the CS paradigm, due to its maturity and ease of management. P2P concepts are used but only sparingly, evidenced by the advent of direct voice-chatting. The impetus to adopt P2P parallels other areas of networking such as Distributed Hash Tables (DHT) and P2P file-sharing. The desired qualities include increased robustness, enhanced resource scalability and of course, load-balancing. However, a P2P-VE introduces specific constraints that places it in a separate class of P2P research.

The focus here is limited to DVEs, where networked users interact with one another in a fictional virtual world. As seen in Figure 5.1, we particularly target FPS games due to their unrelenting and exacting networking demands. They are our primary focus, simply because a workable solution there would arguably work in the more relaxed environments of slower-paced games and VE types.

5.1.1 Voronoi Diagrams

Voronoi Diagrams (VD) are fundamental geometric constructs [Shewchuk, 1999] widely used for spatial partitioning. As seen prior, Fig. 5.2 illustrates a typical 2D variety on the left and a 3D version on the right. It is seen that 2D-VD, when given a set of X-Y coordinates (black dots), neatly partitions the map into polygonal regions. If we imagine those dots as player avatars travelling in a 2D virtual map, it becomes apparent that the

in-game relationships between players/peers are easily discerned. For example, a given player's most immediate neighbours can be seen with relative ease. This is the strength of VDs, their aptness at clustering and deriving the direct communication links between players, as noted in the VON technique [Hu et al., 2006].

The issue of “awareness” [Rueda et al., 2008] or neighbour discovery is a fundamental concern in P2P-VE research. At stake is the connection completeness within the VE, where pools of peers must never be allowed to separate and become unknown to each other. VON confronts this issue via rules that tightly govern the interactivity between neighbouring peers. An example would be the requirement that peers maintain direct links for as long as their Voronoi cells are adjacent, even if they are actually separated by great distances in the virtual map.

A 3D-VD extends the base 2D concept and we see that the regions are now polyhedral 3D spaces (Fig. 5.2, right). While a 2D-VD is not equivalent to a 3D-VD, the ultimate usage is similar. The additional Z -axis is factored in when clustering players, which allows for more interesting linkages otherwise hidden on a 2D plane. It is also used to discriminate between peers, differentiating them on a separate unit of measure.

The novelty of the proposed approach lies in retaining the X and Y axes for in-game

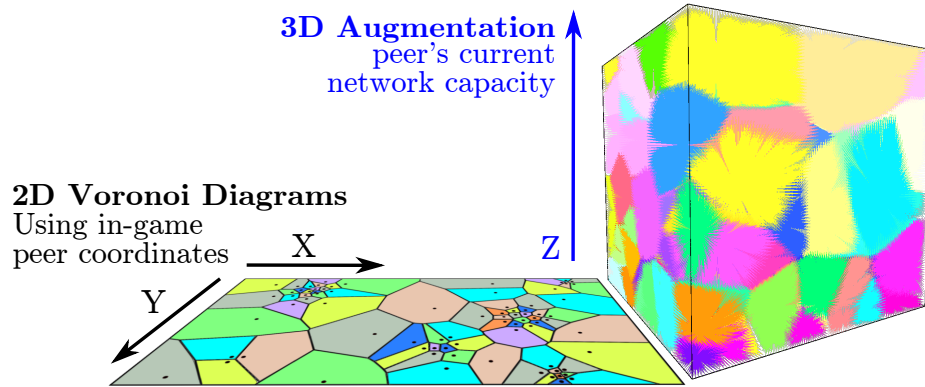


Figure 5.2: A 2D Voronoi Diagram (left) and the enhanced 3D version

coordinates but freeing up the Z -axis to be a non-positional metric. We thus preserve the benefits of prior 2D-VD approaches and gain an additional avenue to tackle the aforementioned load management issues.

5.1.2 Role of arbitrators

The crux of the matter at hand is the selection of dynamic arbitrators in a completely decentralised network. Arbitration services are resultant from the need to resolve peer interactions in a fair manner. For example, if two players (P_X and P_Y) were locked in combat, who decides the ultimate outcome when there are conflicting statements? If P_X says he shot and killed P_Y , and P_Y says otherwise, who are we to trust?

As seen in the Figure 5.3, this is where an independent arbitrator proves useful. It must be noted that this arbitration process happens at an extremely localised, tick-by-tick level. A tick represents a slice of in-game time, a game-turn where players perform iterations of complete moves and have their actions are arbitrated upon. For the VE to appear seemingly fluid, the granularity of a game-turn may be as fine as 33 milliseconds (or lower). This is in order to render the game-world at typical industry targets of 30 to 60 frames per second, which would be visually pleasing to human players.

There are alternative techniques that try to resolve game-play decisions using multiple referees. However, considering that players may easily cheat by modifying outgoing packets [Baughman et al., 2007, Neumann et al., 2007], there is almost always a dispute

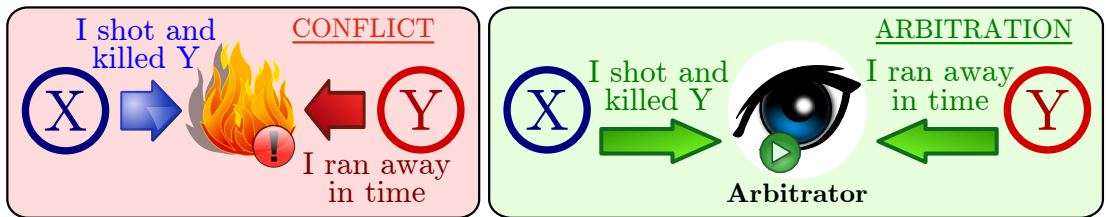


Figure 5.3: Illustrating the inherent conflict of interest in typical FPS interactions and the intended role of arbitrators as dynamic referees.

on outcomes. Thus, it may very well boil down to a single deciding peer, which weakens the case for group voting mechanisms such as [Corman et al., 2006, Webb et al., 2008, Kabus and Buchmann, 2007]. In contrast, our technique borrows heavily from standard industry practices, where singular servers support game-play via lag-compensation methods [Sweeney, 1999, Bernier, 2001]. Vote-tallying or multiple-referee mechanisms are also potentially hampered by the actual time taken to resolve interactions. Often, multiple packets to and fro are needed to set-up, vote on, arrive at and validate any game-play outcome. In latency sensitive FPS games, such delays would prove fatal to game-play fluidity.

This is perhaps why industry has persisted with mature CS paradigms and the singular authoritative server model. Our approach is similar, appointing dynamic game-play arbitrators from amongst the peer population to be quick, one-time conflict resolvers. We make no aspersions regarding the ultimate fairness and security of employing such on-the-fly adjudicators. It is beyond the scope of this paper to discuss the implications of group collusion and reputation management.

5.2 Related Work

P2P-VEs form a distinct class of P2P research, where maintaining timely inter-peer connections is paramount. In contrast, the design of DHTs places emphasis on fault-tolerance and load-distribution. The work of [Knutsson et al., 2004] is a classic example on the implementation of MOGs on P2P/DHT infrastructure. DHTs are used to assign responsibility of in-game objects to peers, similar to how our arbitrators take care of interactions.

However, the emphasis there is on fault-tolerance and load is regarded in terms of processing and storage resources. By comparison, the focus here is on network load, as it is often the most limiting factor in P2P-VE [McCoy et al., 2005, Bharambe et al., 2008]. Further, our arbitrators are highly transient and are not primarily concerned with the long-term storage of in-game objects. Rather, they can be considered as on-the-fly

interaction/conflict regulators.

To enable fully P2P-VEs, a number of techniques have been proposed. They range from using Frontier Sets [Steed and Angus, 2005], constant mutual notifications [Keller and Simon, 2002], recursive map-partitioning [Merabti and Rhalibi, 2004], Delaunay Triangulation clustering [Steiner and Biersack, 2006, Varvello et al., 2007] and area of interest management techniques of [Bezerra et al., 2008, Bharambe et al., 2008]. The defining characteristic of these works has been the need to effectively group peers so as to minimise P2P bandwidth requirements.

Alternative Voronoi-based approaches are found in [Hu et al., 2008, Buyukkaya et al., 2009, Albano et al., 2009], where 2D-VD schemes are augmented in various ways. The issue of load within these works is more related to the distribution and control of in-game objects within the VE. VSO [Hu and Chen, 2011] focuses on the issue of load from an interest-management angle. Using less-frequent partitioning of the 2D map via Voronoi principles, it is concerned with the appropriate matching of events and messages between peers.

In contrast, our focus is on the load incurred when performing game-play arbitration, an issue largely overlooked in prior works. The aim here is for a load-balanced way to dynamically select arbitrators, with our unique 3D enhancement better exposing high-capacity peers within the VE. [Denault et al., 2011] explores interest-management and the load-balancing of in-game objects. However, their partitioning method is entirely static, with provisioned servers in charge of areas that grow or shrink in triangular slices. Further, there is little focus on network bandwidth. [GauthierDickey et al., 2005] tries to limit communication costs by using N-trees to continuously subdivide the game map. However, the focus was on message/event ordering and there is no mention of arbitrators to resolve conflicting updates.

The work contained herein is a significant extension to our earlier efforts in [Almashor

and Khalil, 2010b]. Whereas our preceding focus was on simply reporting on the natural load-balancing properties seen in 3D-VD, this paper is a natural extension and provides:

- More results from the basic use of 3D-VD and a more detailed analysis of the expanded results.
- A further augmentation to counter the phenomena of in-game player flocking and activity hotspots.

5.2.1 Player Flocking Mechanisms

Player flocking remains a popular topic in games related research. Preuss *et al.* in [Preuss et al., 2010] addresses issues when players group up and travel together, although only from a purely artificial intelligence perspective. We imply a more extreme level of flocking, with larger peer counts and are concerned with their implications for P2P-VEs. They form *hotspots* of activity that if left unchecked, would have negative implications. The approach in [Krause, 2009] espouses adaptive ways to reduce the radius of a peer’s interest sphere and thus reduce loads, which may not be ideal for all game-types and does not consider a more fluid use of dedicated server resources.

Our work is similar in scope and aim to Chen *et al.* in [Chen et al., 2005] where the idea is to do away with static map partitioning of the VE, in favour of more dynamic designs. However, they still employ semi-rigid zone assignments to provisioned servers. In the same vein, [Ahmed and Shirmohammadi, 2010] is another body of work aiming for dynamic load management. However, similar to [Chen et al., 2005], they view servers as entities that take charge of load-aggregated, semi-static zones. Their servers are tied strictly to a location, reducing and increasing zone sizes as a function of load.

We take the view that servers are separate, free-roaming entities within the VE that do not actively engage in actual game-play. They bring their resources to bear on any

emerging critical locations. The abstraction here is that we fluidly serve actual peers and not the area constructs that contain them.

5.3 Automatic Load Management

The scenario we envisage is of an FPS game set in a free-roaming virtual world. We assume that heavier game-world assets (sound files, 3D textures, object behaviour code, etc.) are pre-installed, following the typical distribution models of current on-line FPS games. Thus, each and every peer in the VE is self-contained, ready to explore the game-world in all of its entirety. Accordingly, there is no transmission of large files amongst peers. The exchanged data is confined to simple, basic update messages between 40-80 bytes containing the game-moves and status of players, including: current position, direction faced, ammunition levels and so on.

Indeed, the problem is more nuanced as we are not concerned with load-balancing issues relating to heavy data transfers. Rather, the network connections needed by arbitrators to perform their duties are the focus here.

5.3.1 Using 3D-VD

To actually select the arbitrators, we turn to 3D-VD and the Z -axis . Usage remains similar to the VON approach but, as seen in Fig. 5.4, we see how the 3rd dimension allows a more natural method of neighbour (NB) classification. At this stage, we simply order a node's neighbours based on their current loads (see ranking table in Fig. 5.4), which provides a seamless way to both connect all peers and expose certain types of amongst them. For example, high-capacity peers would be more visible to others, via the Z -axis . With standard 2D-VD, such peers would likely be occluded, effectively blocked from view by intermediary nodes on a 2D plane.

3D-VD, being a modified extension of VON, would maintain connectivity on a global

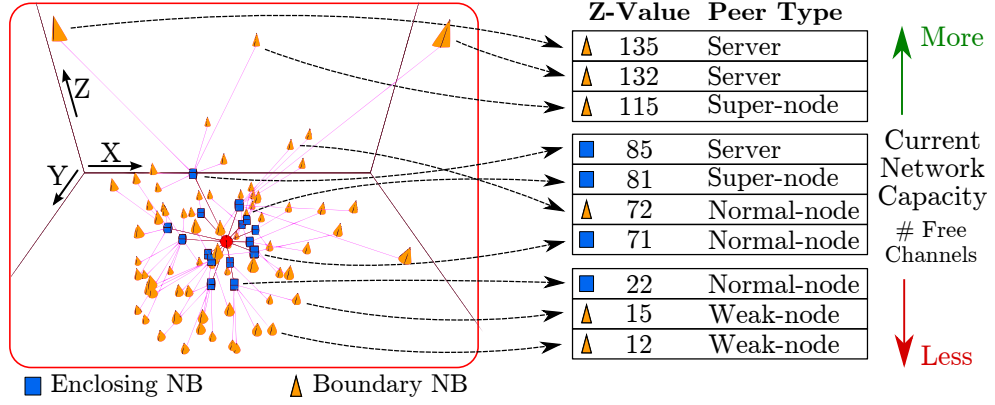


Figure 5.4: Each peer derives its own 3D-VD view of the world around it, decides on its neighbours (NB) and subsequently ranks them in terms of arbitrator suitability. This happens **at every tick** of the simulation, for all peers. [Almashor and Khalil, 2010b]

scale in the face of any network volatility. By ensuring each peer connects to the right neighbours at a localised, per-node level, we ensure link completeness on a global scale [Hu et al., 2006]. The technique differs slightly from VON, as the peer connectivity rules are simplified. Hu *et al.* were concerned with scalability issues and used the concept of dynamic, circular Areas of Interest (AOI) around peers to potentially reduce the number of required connections.

With our focus on the issue of arbitrator load balancing, we have opted to remove the AOI mechanism. Instead, we simply made peers connect to (i) all of their closest neighbours (enclosing peers with adjacent polyhedral Voronoi regions – blue cubes in Fig. 5.4); and (ii) all immediate neighbours of those enclosing peers (the outer yellow cones). Thus, given a computing peer P_C , the enclosing neighbours are the first tier, forming a tight envelope around it. Subsequently, the second tier of boundary neighbours form an all-encompassing layer around P_C and its enclosing neighbours.

This modification allows us to isolate experimental variables and better evaluate 3D-VD's ability as a load-balancing mechanism. We safely generate a single global 3D-VD that is then referenced by individual peers. Given the assumption of an ideal setting

where no peers disconnect, the complexity of the simulation engine is thus reduced and experimentation eased.

5.3.2 Defining Load

When appointing an arbitrator, it is naturally assumed that the fittest candidates are (i) nodes that are lightly loaded and (ii) nodes that are in or around the vicinity. A less-loaded peer would have the necessary resources to assume the role (which involve extra processing and communication obligations). A close proximity peer would already have vital localised data, leading to reduced set-up costs. There would be minimal pre-processing, such as the loading of static in-game objects.

While discussing the term load on such systems, it is important to note the most vital constraint in MOGs: network resources. In recent times, storage and processing have diminished in importance, due to technological advances. In contrast, bandwidth for typical household connections have remained relatively stagnant. In fact, we need to consider an even more limiting sub-factor of current generation network resources, that of upload bandwidth. Typical home connections, which represent our main application area, have upload requirements that are usually limited to 1Mbps [Bharambe et al., 2008]. This ultimately restricts any given peer's ability to provide arbitration services to fellow VE participants.

Thus, the load-balancing discussed in this paper is not to be construed in the classical sense. As seen in Figure 5.4, the differentiation along the Z -axis is based entirely on current upload capacities of peers. Accordingly,

- It becomes easier to select a suitable, **resource-available** arbitrator in a **timely** fashion.
- The system is enhanced as an even more natural load-balancing tool that directs load **appropriately**.

Appropriate distribution, within this context, represents a mechanism to intelligently direct loads. With a population of peers made up entirely of normal nodes, we would just select any peer that happened to be non-loaded at the time. However, when super-nodes with expanded capabilities are present, then these peer types should be regularly appointed as arbitrators.

The load-balancing algorithm can thus be conceptualized as a two step approach: We first utilize 3D-VD to cluster and rank the neighbours of a node. Then, we employ the simplest strategy of always selecting the neighbour highest along the Z -axis. For comparisons, we implemented alternative strategies as well, ranging from prioritizing based on euclidean distances (in both 2D/3D space) and what we term mass-effect fields using Newtonian laws on gravity.

However, it was discovered that randomised selection, where the peer randomly chooses a neighbour, performed best of these alternative policies. Thus, for brevity, we only report on results from two policies, namely: Fittest Arbitrator Selection (FAS) and Random

Algorithm 2: Arbitrator Selection at each tick

```

1  $PosnUpds \leftarrow$  from all nearby peers;
2  $VD \leftarrow \text{ComputeVoronoi}(PosnUpds)$ ;
3 foreach Candidate Peer,  $P_C$  in  $VD$  do
4   if  $P_C$  is a neighbour (enclosing or boundary) then
5      $\mid$  Add to  $Ranked_{NB}$ ;
6      $\mid$  Send Reciprocal Update;
7   else
8      $\mid$  Drop from  $Ranked_{NB}$ ;
9 if Combat Initiated then
10  switch Selection Policy do
11    case FAS
12     $\mid$   $Arbi \leftarrow Ranked_{NB}[1]$ 
13    case RAS
14     $\mid$   $Arbi \leftarrow Ranked_{NB}[\text{Random}()]$ 

```

Table 5.1: Peer Configuration Codes and Details

Configs	Servers	Super nodes	Normal Nodes	Weak Nodes	Remarks
00-00-00-X0	0%	0%	0%	100%	All Constrained (weak-nodes only)
00-00-50-50	0%	0%	50%	50%	Even mix of normal and weak-nodes
00-00-X0-00	0%	0%	100%	0%	Egalitarian (all normal-nodes)
00-10-45-45	0%	10%	45%	45%	Enhanced with super-nodes
00-10-60-30	0%	10%	60%	30%	Super-node Enhanced (majority normal-nodes)
10-00-45-45	10%	0%	45%	45%	Enhanced with servers
10-20-50-20	10%	20%	50%	20%	Fully enhanced
s1-00-00-X0	1	0%	0%	100%	Single Server with All-Constrained
s1-00-50-50	1	0%	50%	50%	Single Server with 50-50% mix
s1-00-X0-00	1	0%	100%	0%	Single Server with Egalitarian
s4-00-00-X0	4	0%	0%	100%	Four Servers with All-Constrained
s4-00-50-50	4	0%	50%	50%	Four Servers with with 50-50% mix
s4-00-X0-00	4	0%	100%	0%	Four Servers with Egalitarian

(RAS).

Algorithm 2 summarises the selection process that is performed at each simulation *tick* (a slice in simulation time). This process is repeated for each and every peer in the virtual world except for the servers, who are not expected to engage in game-play and thus have no need for arbitration services themselves. The approach is simplistic by design, as we intend for our work here to form the basis for future enhancements that consider other VE parameters. Nevertheless, as seen in the analysis of simulation data, such straightforward

selection mechanisms already exhibit strong load-balancing properties.

5.3.3 Using the Z -axis

The scale of the Z -axis plays a significant role in determining the resultant 3D-VD. For example, derivations will change when the Z -axis goes from a range of 0 to 100 to say, 0 to 1000. This is due to the underlying Delaunay Triangulation (the dual structure of VD), where the angles forming a triangle (or 3D tetrahedrons) tend to be as obtuse as possible. We address this issue by normalizing the 3rd dimension between predefined minimum and maximum values across all peers, resulting in a standardized Z -axis.

It must be noted that there is a distinction between the maximum values for particular types of nodes and the overall Z -axis maximum. A peer's maximum capacity only relates to itself and is a direct function of its type. Thus, a constrained (weak-node) peer can never reach as high as typical peers (characterised as normal-nodes) on the Z -axis. Likewise, normal-nodes never reach the levels of super-nodes, and so on.

However, movement downwards to the minimum Z value is entirely possible. A server (and every other type of peer) may reach the lower levels of the Z -axis and become on par with weak-nodes. This happens as a peer becomes ever more loaded due to ever increasing demand for its arbitration services. For our experimentation, the global maximum is taken from the maximum values of nodes denoted as servers and all other peer types have their Z values normalized against this.

For simplicity, the following maxima were used for each peer type: **Weak Nodes** : 50% of the capacity of a Normal Node; **Normal Nodes**: reference value (100%); **Super Nodes**: 150% of normal capacity; and **Servers**: 400% of normal capacity. For example, should we assign the value of 1 Mbps to a typical (i.e, normal) node, the servers would have a 4 Mbps upload capacity.

5.4 Experimental Results

There is a need to test 3D-VD across a range of network configurations consisting of differing distributions of peer/node types. This was in an effort to expose the impact of using our modified Z -axis. Table 5.1 lists the various topologies involved and details the percentage distribution of peers in each. It also contains the codes used to signify their respective peer configurations, used throughout the plots presented later.

Each double digit indicates the percentage distribution of a particular peer type. As seen in the table, the code “10-20-50-20” refers to a topology with 10% dedicated servers, 20% super-nodes, 50% normal-nodes and 20% weak-nodes. “X0” indicates a 100% distribution, meaning all peers are made up of that particular type. When paired with an initial “s1” or “s4”, one or four peers have been made into dedicated servers.

Of the minimal-server configurations, sole servers were placed in the middle of the virtual world. In configurations with four servers, each were placed in a separate quadrant. In the server-enhanced topology with 10% of peer population converted to servers, the servers are placed randomly. All servers were stationary throughout the simulation runs.

Due to the large number of peers, no actual inter-client testing (with data delivery over a network) was possible. For each tick within the simulated VE, a single, global 3D-VD was derived, with each peer subsequently carving out its own subset 3D-VD to discern its neighbours. This was purely a performance consideration, to avoid slow derivations of thousands of 3D-VDs on a single machine. 3D-VD computations were made possible through the use of CGAL [Pion and Teillaud, 2009], a computational geometry software library.

Note that no actual game-play mechanics were simulated beyond the in-game movement of peers, as such details do not contribute towards our investigation of the network bandwidth loads of arbitrators. Peers move randomly within our system and when triggered, selects an opposing peer and an arbitrator. The simulation of interaction/combat is

simplified to the creation and maintenance of additional links between the combatants and the arbitrator, sustained over 4 simulation ticks to represent increased networking loads for the arbitrator.

Of the results, there are three key areas of analysis. Each is indicative of the average values obtained by a peer *throughout its life* in the 3DV simulation, namely:

1. **Z-axis Position:** Referring to the average number of free channels a node records throughout its life.
2. **Z-axis Fluctuation:** Average fluctuation in the number of free channels experienced by peers.
3. **Number of Neighbours:** The average number of neighbours a peer is required to connect to.

5.4.1 Average Free Channels (AFC)

The term “free channels” refers to the individual upload links that a peer has available, at each simulation “tick” (i.e., a single turn or a slice of time in the virtual-world). At each tick, a peer can maintain communications with a number of its neighbours, up to a preset maximum defined by its type (as shown in Figures 5.5a and 5.6a with the vertical lines signifying maximum values for weak-nodes, normal-nodes and super-nodes). The number of free channels is thus the number of unused upload links after a peer has sent its update messages. The more free channels, the higher its current capacity and thus, the higher its position on the Z -axis . If it happens that the peer is heavily loaded (with neighbour connections or by arbitration duties) or is a naturally constrained weak-node, then it will consequently and consistently be lower on the Z -axis .

The average position on the Z -axis is, in essence, a peer’s overall height in the virtual cube. It is reflective of the average number of free channels that a peer has throughout

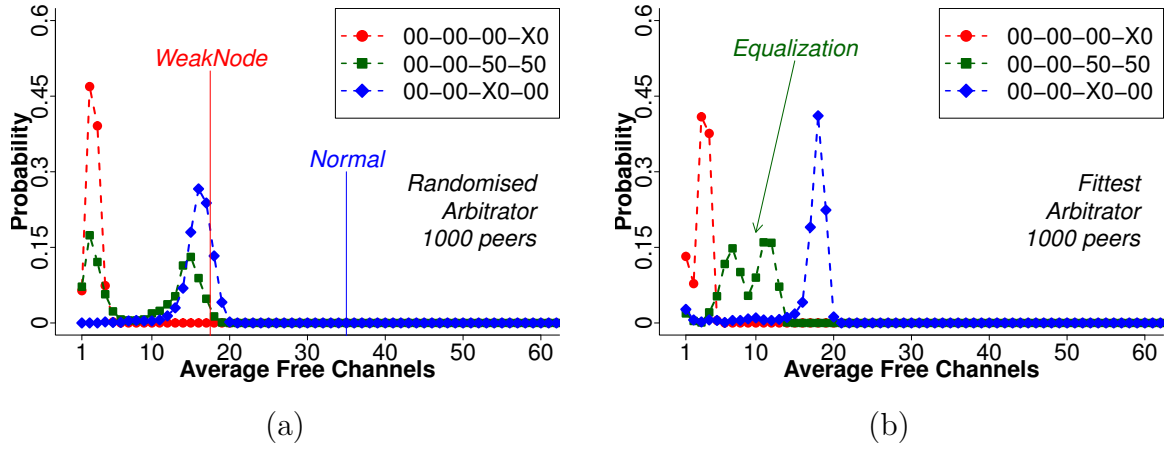


Figure 5.5: The probability mass functions of the Average Free Channels (AFC) per peer, which are analogous to the average positions of peers along the Z -axis. The plots are for a configuration of 1000 peers with egalitarian make-ups (i.e., no servers or super-nodes). (a) shows performance with a randomised arbitration policies, whilst (b) shows the fittest arbitrator policy in action.

the simulation run. Referring again to Figures 5.5, 5.6 and 5.7, the X -axis of all plots are analogous to the Z -axis in the 3D-VD (imagine the vertical Z -axis made to lie horizontally).

Studying the average height of peers in the simulation provides insights into the natural processes of a 3D-VD. This is perhaps the most significant indicator of load-balancing properties. Available resources are the direct inverse of the load a peer experiences. In any P2P system, the desire is always to maximise available resources and effectively spread the load. Similarly within our context, we wish to distribute the network load evenly amongst peers and at the same time, utilize any high-capacity resource points effectively.

This is aptly evidenced in Figure 5.5b, where it is seen that using a **Fittest Arbitrator Selection** (FAS) scheme results in the equalization of AFC values for the 50-50% mixed configuration. The probability mass function (PMF) show two slightly overlapping peaks. The left peak represents the majority of weak-nodes and the right is indicative of the Z -axis locations of the normal-nodes. They are situated in between the two higher, sharper peaks characterising configurations where all the nodes are weak and where all the nodes

are normal, respectively. Contrast this with the plot shown in Figure 5.5a, where **Random Arbitrator Selection** (RAS) is used. We can see the peers in 50-50 mixed configuration behaving in a similar manner to its pure counterparts.

The reasoning behind the equalized plot in 5.5b is that the normal nodes, being naturally more freer, are regularly being selected as the arbitrators and are subsequently *pulled left* (towards the weak nodes). In contrast, the weak nodes, having been mostly relieved of arbitrator duties, enjoy greater AFC values and thus *move right*. This indicates a good spread of load, as the normal nodes are being made to help weaker ones.

This feature is also exposed in the super-node enhanced configurations seen in Figures 5.6a and 5.6b. This time, the total population is doubled to 2000 peers and we see that with RAS enabled, there exists a concentration of super nodes around the 20 to 30 AFC marks. This is where the 10% distribution of super nodes are situated. Subsequently, when FAS is turned on in 5.6b, the super nodes are shifted left and are merged to form a sharp peak with the normal nodes. Also note the higher values obtained by the weak nodes, as

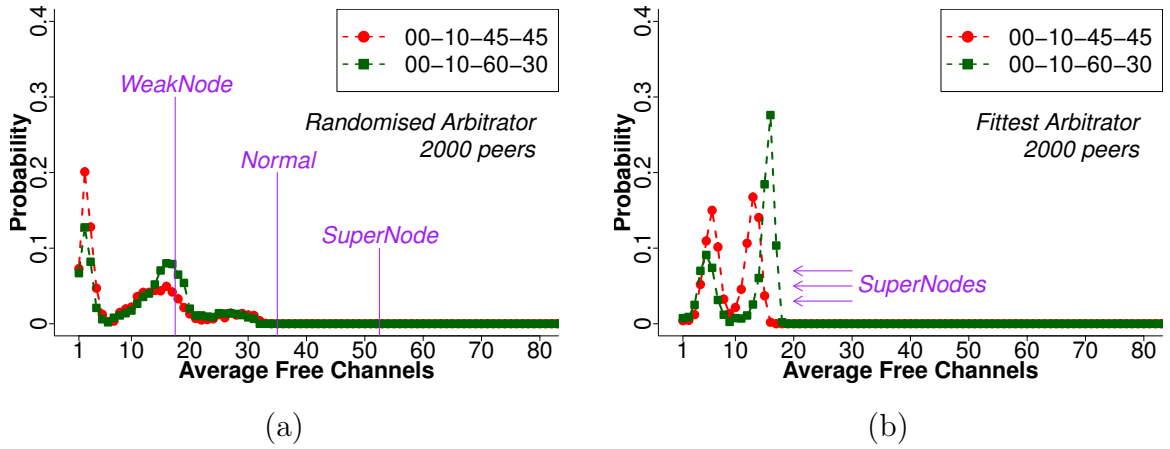


Figure 5.6: Similar to Fig 5.5, the plots here show the PMFs of per peer AFCs. These have configurations of 2000 peers that are enhanced by **super-nodes**. (a) shows performance with a randomised arbitration policies, whilst (b) shows the fittest arbitrator policy in action.

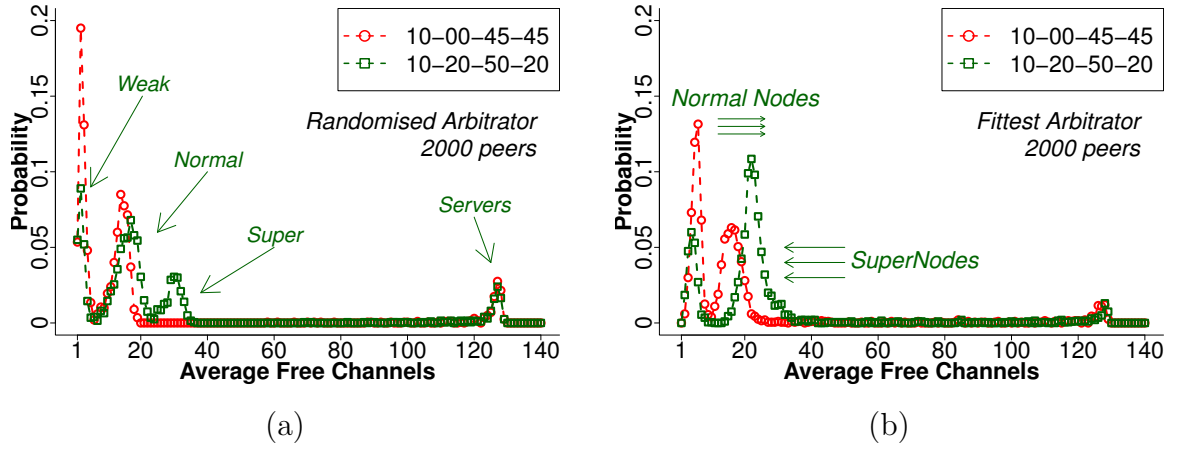


Figure 5.7: PMF plots of per peer AFCs for configurations of 2000 peers with enhancement by **servers** this time. **(a)** shows performance with a randomised arbitration policies, whilst **(b)** shows the fittest arbitrator policy in action.

they are helped by a combination of normal and super nodes.

Finally, in Figure 5.7a, we see the individual peaks for concentrations of weak, normal, super nodes and servers from left to right. It is thus revealed that RAS does not effectively utilize servers and super-nodes, who respectively form 10% and 20% of the peer population. Note how the peaks are clustered around each peer type, demonstrating an inability by higher-capacity peers to help those below. For the 10-00-45-45 configuration, the servers are largely wasted. In contrast, we see how FAS works to alleviate the network load experienced by weaker nodes in Figure 5.7b. Note how the servers are utilised more, as evidenced by a smaller peak on the right.

5.4.2 Fluctuations in Free Channels

In the prior subsection, we investigated the average number of free channels (AFC) a peer has throughout its life in the simulation. AFC is a good approximation of a peer's expected position along the Z -axis. Here however, we study the *fluctuations* a peer experiences along the Z -axis, which effectively means the fluctuations in the number of free channels.

Fluctuation offers indirect evidence of a peer’s utilisation levels as it engages in the VE. All peers will experience a degree of fluctuation in the number of free channels it has. As it moves around, a particular peer discovers new neighbours, loses contact with old ones and may be chosen as arbitrators. Such events contribute to movement along the Z -axis.

For example, if a peer happens to be of a type server, it may eventuate that 100 normal-nodes will all request arbitration services from it, all at the same time. This will cause it to rapidly drop down in terms of its free channels count. As it recovers from the load, it will float upwards along the Z -axis, as more and more of its available links become free. It is the average fluctuation (AFLUX) that is of interest here. High AFLUX values indicate a peer that is constantly “yo-yoing” along the Z -axis. Such behaviour is undesirable as:

- It is harder to engineer a dedicated server’s requisite capacity. In all likelihood, over-provisioning is necessary and server resources are thus wasted.
- It exposes the inefficient use of available resources. Balancing the load amongst peers is one desirable outcome, effectively directing the load to appropriate peers is another

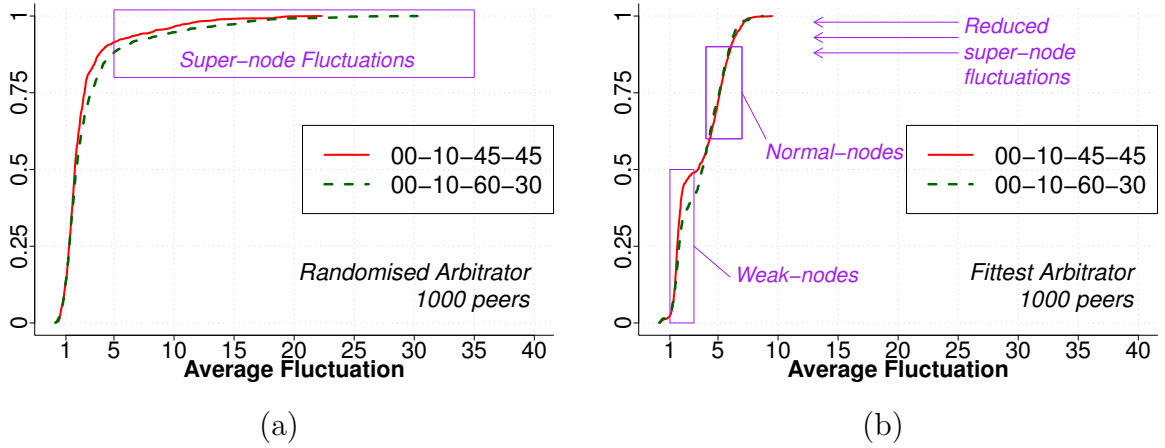


Figure 5.8: Cumulative distribution functions (CDF) of the average free channels fluctuations (AFLUX) experienced by peers. Shown here are configurations with 1000 peers containing 10% super-nodes.

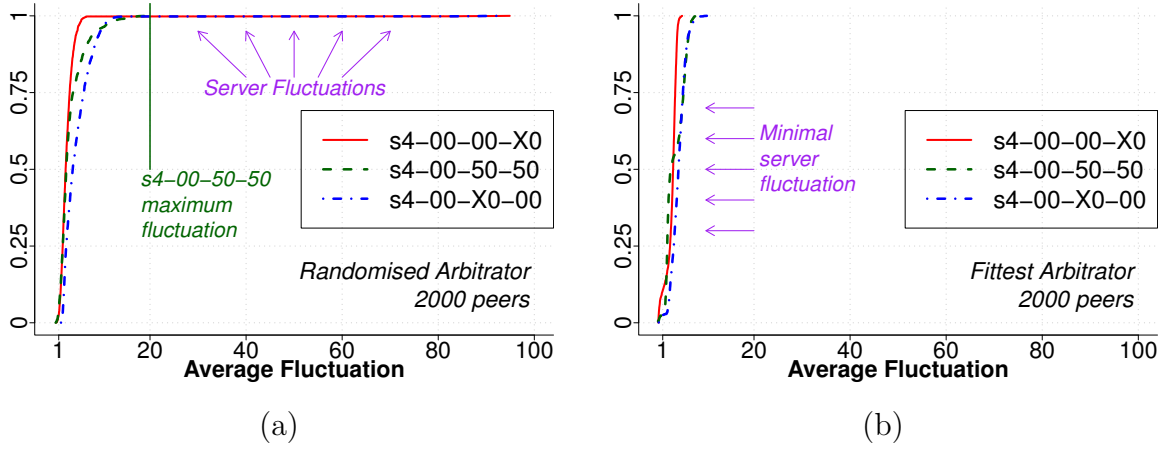


Figure 5.9: CDFs of the peer AFLUX values. Shown here are configurations with 2000 peers containing exactly 4 high-capacity dedicated servers.

more significant feature.

In contrast, low AFLUX values would be symptomatic of peers having a constant amount of load through out its life in the simulation. This is especially significant when we consider the super-nodes and dedicated servers in our system. These are the peer types that have more resources and appropriately, they need to be utilised more, in order to help the constrained peers.

This is evident in our 3D-VD system as we study the CDF plots in Figures 5.8a and 5.8b. For the RAS results, notice the elongated high, horizontal lines, indicating that a subset of the peers experience high fluctuation levels. Closer inspection of the raw data confirms that it is the super-nodes that are registering these high values. Contrast this with the results shown in 5.8b, where FAS is used there exist less overall AFLUX values. The conclusions to be drawn are that the super-nodes are being regularly (and appropriately) utilised as arbitrators, maximising their utility to the P2P-VE.

This outcome is also seen in Figures 5.9a (where RAS is used) and 5.9b (FAS). The comparison is based on the specialised 4 server configurations (i.e., s4-xx-xx-xx). It is seen

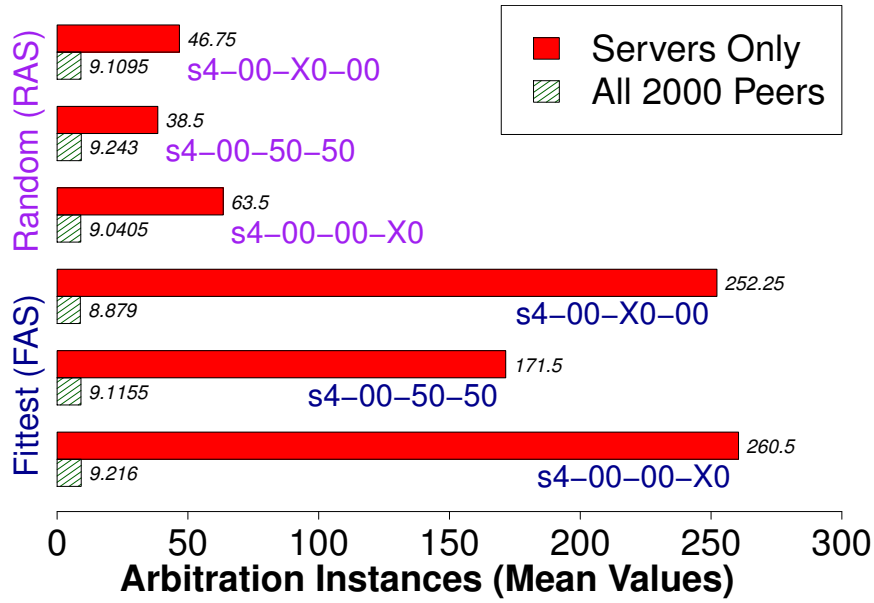


Figure 5.10: Comparison of server workloads between Random Arbitrator Selection (RAS, top) and Fittest Arbitrator Selection (FAS, bottom) policies. Configuration with 4 servers are shown as none of the other peer configurations contained servers.

that the same high fluctuations are seen in RAS and then successfully stymied in FAS. Again here, high-capacity peers (i.e., servers) are constantly being used as arbitrators, as they should be.

More insights can be found in the mean values for arbitration instances, as seen in Figure 5.10. Arbitration instances refers to the number of times a peer is called upon to provide arbitration services and it is seen that FAS induces the four servers in question to assume this role much, much more than the population average of ~ 9.0 . This is why they are more stable in FAS; they are simply being utilized in a constant manner, which curtails fluctuation.

A mistake would be to infer that the four servers are taking the lion's share of the arbitration loads. As made apparent in Figure 5.11, this is shown to be false. Here, we see that when using FAS, they only take up a small slice (roughly 5%) of **18000** total

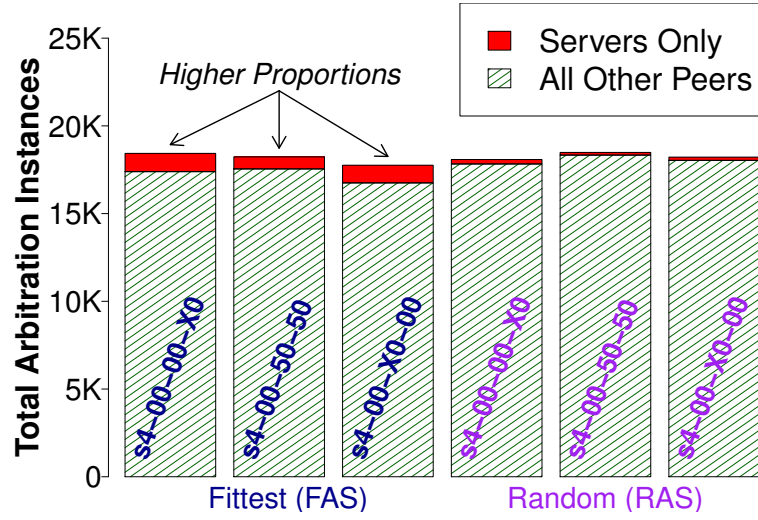


Figure 5.11: Shows the proportion of arbitration workloads that servers process in relation to the rest of the peer population. The FAS policy is shown on the left, whilst RAS is shown on the right. Again, only server-enabled peer configurations are included here.

arbitration instances. While it is significant that four peers are doing 5% of the workload in a 2000 peer configuration, it does demonstrate that servers are not overloaded. In contrast, we see the much lower usage levels when using RAS. The servers are simply being inconsistently used, which inadvertently leads to high AFLUX values.

Note however, the higher mean values recorded for servers even when using RAS, which are still higher than the peer average. This phenomena is a function of the 3D-VD. As servers are not always selected as arbitrators (in contrast to FAS), they stay higher up on the Z -axis with more free channels. By staying above the rest of the peers, they are also exposed to more peers (fellow normal and weak nodes). This exposure induces them to maintain links with far more neighbours and leads to more likely selection, simply due to their larger number of neighbours.

5.4.3 Number of Neighbours

Another key indicator of load is the Mean Number of Neighbours (M-NNB) a particular peer connects to as it moves around. It is effectively the number of links that a peer must maintain at any time. As the Z -axis metric is a peer's network upload capacity, and as we are interested in load-balancing, M-NNB offers a compelling perspective to our issue.

It can be easily seen that higher M-NNB values leads to a higher demand for upload bandwidth at a particular peer, essentially making it increasingly loaded. Thus, the aim is to minimize the M-NNB values, as doing so places a lesser burden on each peer's network upload requirements. Yet, this is a challenging issue as the very nature of a 3D-VD induces more exposure amongst peers to each other, simply via the introduction of the Z -axis. Peers are more likely to "see" each other due to this added visibility and thus, have to maintain more links in order to secure the consistency of the P2P-VE.

In contrast, a 2D-VD will naturally occlude subjects that are just beyond a peer's immediate circle. An example of this would be a full solar eclipse: where someone within the umbra has his view of the sun completely blocked by the moon during such events. If we imagine all the celestial bodies on a single 2D plane, it becomes apparent that there is less freedom of interaction between the earth (a peer) and the sun (a distant fellow peer) due to the presence of the moon (an immediate neighbour). Utilize a 3D-VD however and suddenly, we have a whole new axis with which to be exposed to the sun.

We see this neighbour multiplying effect when we refer to Figure 5.12. The graphs show M-NNB values obtained for 1000 and 2000 peers, respectively. The average of M-NNBs across all peers (i.e., the mean of means) hovers around the 75 to 100 marks, indicating that on average, a peer needs to maintain links with up to 100 other peers, in all simulation configurations. Such values are in fact, relatively high, especially when we consider the limited upload bandwidth of typical home broadband connections. Even at modest 80 bytes per update sizes, sending them to 100 peers at a rate of 30 updates per second will

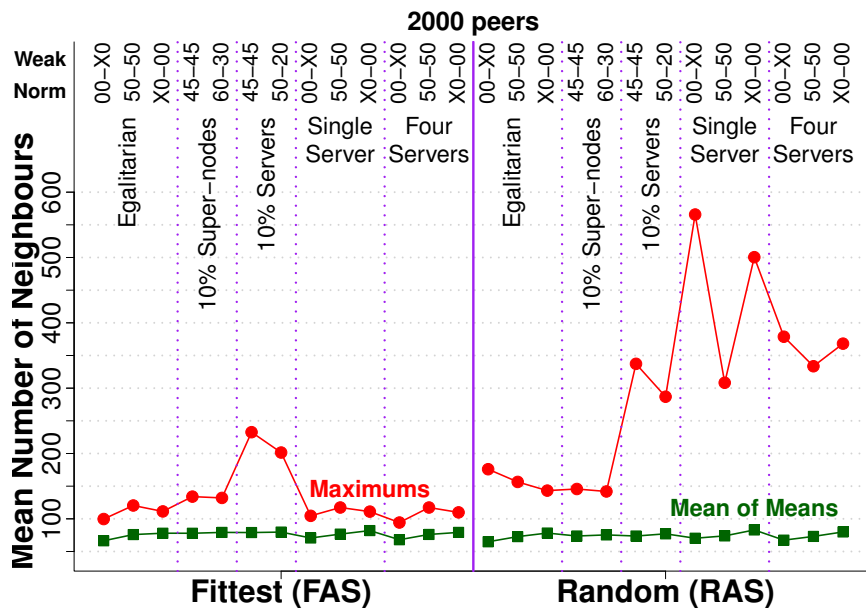
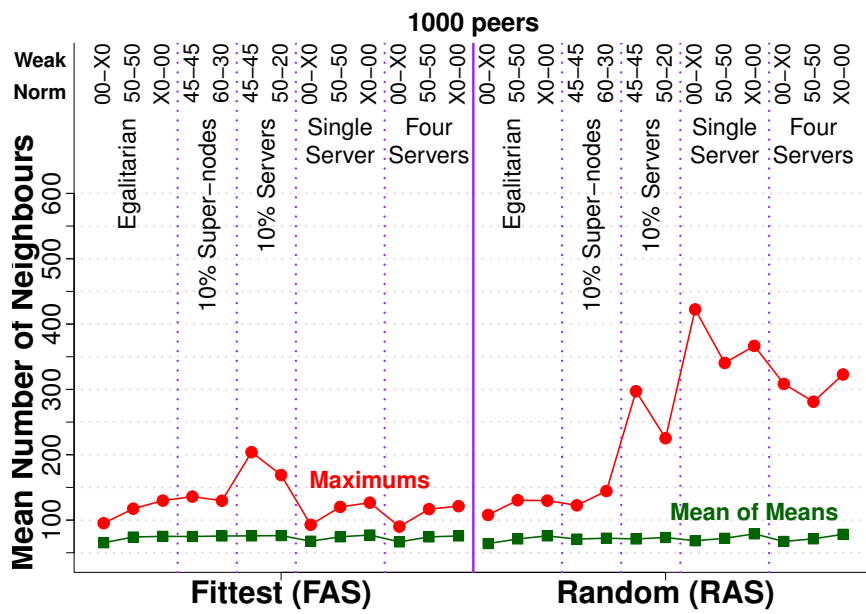


Figure 5.12: The Mean Number of Neighbours (M-NNB) across all configurations are displayed here. On the left of each plot shows results when FAS has been activated, whilst RAS is shown on the right.

quickly overload peers. This is a fundamental issue with 3D-VD and steps were taken to address it in [Almashor and Khalil, 2010a]. While it is an interesting conundrum, we hesitate to discuss it at length within the context of this paper.

One other particular note of interest is the relative stability of the M-NNB across all configurations, as evidenced in the lower plotlines (squares) seen in both Figures 5.12a and 5.12b. The choice between the competing FAS and RAS policies is made less obvious here. However, this stability in mean values belies hidden characteristics that are only exposed when we study the *maximum* M-NNB values recorded. We see at once that there are at least one peer in each configuration that records higher values, particularly in server enhanced configurations (i.e., sv4-xx-xx-xx, 10-00-45-45, etc.) and particularly for RAS schemes. One would, by now, hazard a guess and claim that the high values were registered by the servers. Further scrutiny of the raw data proves this to be a correct assumption.

Similar in fashion to how dedicated servers experience much higher degrees of Z -axis fluctuations in the prior subsection, we see that servers here are penalised with high M-NNB values when using RAS. As can be seen in Figure 5.12b, there is an instance where the lone server in the s1-00-00-X0 configuration is forced to connect to over 550 other peers. Disregarding any current arbitrator duties, that effectively means that at one stage in the simulation, the server sent 352000 kilo-bits of data in one go ($550 \times 80 \text{ bytes} \times 8$). That is slightly over a third of a typical 1Mbps broadband upload bandwidth used, all just in an effort to update its position to fellow peers in the 3D-VD. Considering that we strive to maintain a 30 update per second (to match a 30 frames per second rendering rate), the calculated figures are unnecessarily high. Although it may not exceed a dedicated server's upload capacity, it is nonetheless, an ineffective utilization of resources.

The cause of this phenomena lies in the nature of a 3D-VD, and how it imposes linkages on exposed peers. Figure 5.13 is an illustration of this effect. On the left, we have two servers (square and circle) with currently high capacities floating near the top. The normal

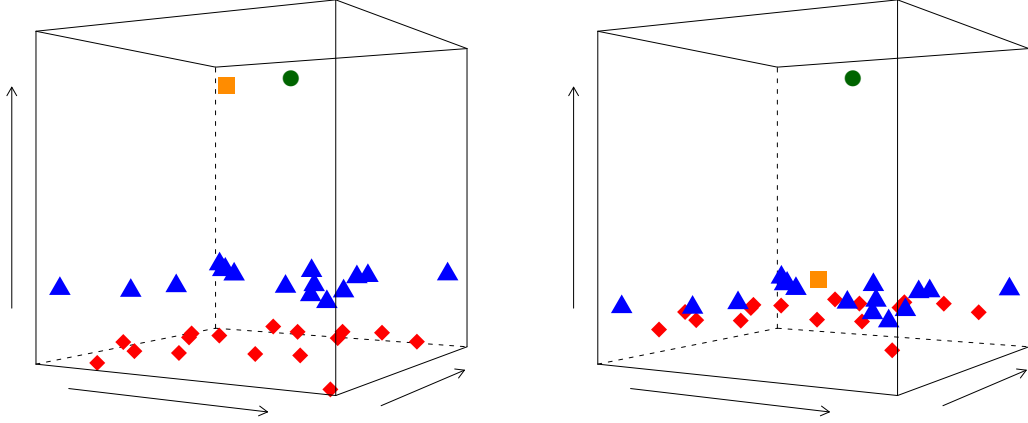


Figure 5.13: Illustrating the effects of **peer layering** and **exposed locations** in a 3D-VD. Server-1 and Server-2 are the square and dot, respectively. Normal-nodes are triangles with diamonds being weak-nodes.

nodes (triangles) form a dense layer beneath them with the weak-nodes (diamonds) forming a further substratum. Recall from our Z -axis position (AFC) findings, that this topology would naturally occur (refer Figure 5.7a). In this scenario, the two servers are exposed to mainly the normal nodes, while a majority of weak-nodes are excluded.

On the right, we see three events: (i) Server-1 (the square) has been pulled down alongside the normal-nodes; (ii) the normal-nodes have collectively been lowered slightly; and (iii) weak-nodes have risen to almost the same tier. As mentioned prior, events (ii) and (iii) are the result of applying FAS mechanisms (equalization in Z -axis positions).

With reference to the servers, two important after-effects are thereby demonstrated. First, Server-1 now has less NNB to connect to as it is roughly on the same plane as the other peers. The nodes around it seemingly form a protective shell, thus alleviating any necessity on its part to contact nodes along the fringes. This is why it is desirable to pull servers (or super-nodes) down; doing so effectively relieves some of their updating burden

and frees up channels for arbitration purposes.

In contrast, Server-2 is a demonstration of the causes of the issues with RAS mechanisms. As it remains high up along the Z -axis (because RAS policies are less likely to select it as an arbitrator), it is far more exposed to all the other peers. When coupled with a flatter layer of a mixture of normal and weak nodes, it proves to be a potent combination that negatively affects the load, status and utilisation of Server-2. As can be extrapolated from the right diagram, Server-2 is potentially neighbours with *all* other peers. As RAS is also less likely to pull it down, it stays up high on the Z -axis and thus, makes it waste resources just by maintaining contacts with more peers.

Server-2 is also greatly susceptible to dramatic fluctuations. The more neighbours it has, the more the likelihood of these neighbours *all* randomly choosing it as an arbitrator. Thereafter, they may abandon it as an arbitrator, thus causing the “yo-yoing” effect we see before. These occurrences becomes ever more likely as we add more and more peers to the population, further exacerbating the fluctuations we observe.

Figure 5.13 also provides an explanation to the curious swings in maximum values for the single server configuration in Figure 5.12b (and to a lesser extent, the s4-xx-xx-xx configurations as well). On the left, when the normal and weak nodes form two separate tiers, the servers have only half of their fellow peers to contend with. This translates to the troughs associated with s1-00-50-50 and s4-00-50-50 configurations. The cause is that they simply have fewer number of adjacent neighbours.

Again drawing back to Figure 5.12, we see that with the weak exception of configurations with 10% server distributions, FAS is decidedly better at minimizing the M-NNB of high capacity peer (i.e, servers and super-nodes). Even in those 10-xx-xx-xx configurations, FAS is able to reduce maximum values by a factor of around 100 peers, a good improvement over RAS.

The final point of interest with Figure 5.12 is the relatively high maximum values

seen for 10% Server configurations, even in FAS scenarios. The cause here is simply an overabundance of servers; the normal and weak nodes are spoilt for choice when obtaining an arbitrator. The workloads generated are comfortably handled by the 100 available servers (200 for populations with 2000 total peers). As a result, not all the servers are being utilized.

Consequently, servers stay higher up on the Z -axis, are subject to more neighbours and thus, incur more link maintenance costs. In retrospect, blindly increasing the amount of dedicated servers is not entirely beneficial. In fact, server resources are squandered on maintaining trivial updates. The conclusion here is that 3D-VD can tell when classic over-provisioning occurs and with the aid of FAS policies, help address it.

In summary, the benefits of low M-NNB values are:

- It indicates that servers are being pulled down into the layers of the (relatively) weaker nodes, which is evidence of them being utilised consistently.
- By being in and around the same level, servers limit their NNB requirements, which means they are not wasting resources by sending simple update messages to hundreds of peers. Instead, available up-links are being used to fulfil arbitration duties.

5.5 Flocking and Hotspots

With the core 3D-VD technique, it was demonstrated that basic load-management issues within a P2P-VE could be suitably addressed in an automatic manner. The next step tackles concerns relating to unpredictable and potentially detrimental scenarios, such as **player flocking** [Preuss et al., 2010].

Flocking or crowding is a naturally occurring phenomena, mirroring human behaviour in the real world. Its manifestation within DVEs results from the tendencies of players to gravitate towards each other and towards any events occurring within the virtual world. It

is indeed a variant of the well-known “flash crowd” problem. Both feature sudden spikes in demand for resources, leading to overloaded servers and disruptions.

An example of flocking would be a battle between two opposing sets of players. “Flockers” form a sub-set of the peer population and the venue where they congregate is termed the event’s *hotspot*. This is typically an area of heightened activity as players interact more with one another. Such events are typically hard to predict and simply over-provisioning high-capacity servers at specific VE areas is an imprecise, inefficient and inflexible solution. Correspondingly, the networking requirements for these increased interactions stresses the capabilities of servers in the area. As an example, a server is tasked with handling a specific area in the virtual world (e.g., a house). Subsequently, an event occurs and many players gather at that location (e.g., a meeting called by a player).

Consequently, the server is unable to service all mediation requests, possibly stalling ongoing peer interactions. This would be fatal to the fluidity of the VE. Fast paced games like FPS, in particular, would undoubtedly suffer with game-play experience degraded. In such demanding applications where reactions are measured in the range of milliseconds, any delays to the start of combat must be avoided.

Artificially partitioning the VE and simply over provisioning servers is imprecise and inefficient. P2P-VEs also connote full decentralisation, with no mechanisms to collect system runtime data. As such, global knowledge cannot be relied upon to detect and then allocate servers to troubled hotspots.

Thus, to address the issue of player flocking in a fully decentralised VE, we need mechanisms that:

- allow servers to roam freely within the virtual map, unrestricted by arbitrary partitioning schemes
- detect growing hotspots and disseminate news of their presence to servers in a timely fashion

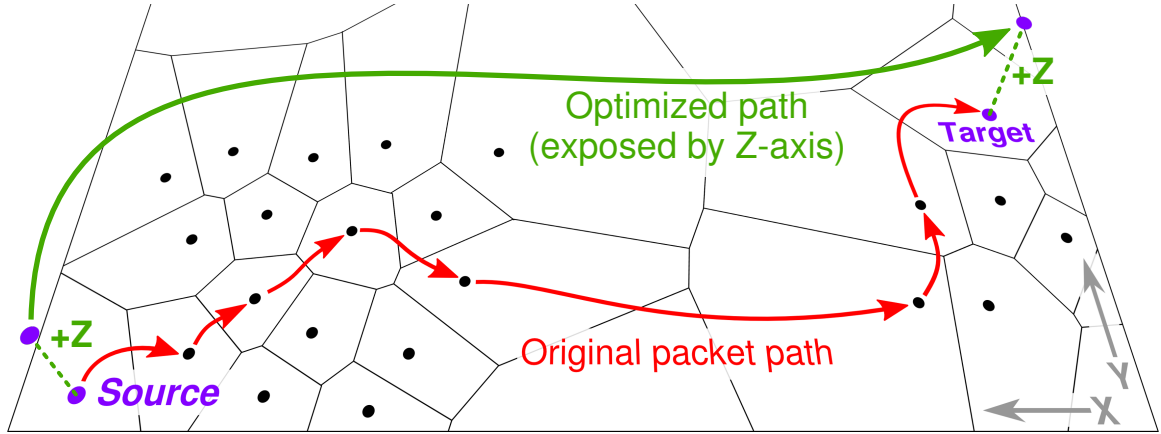


Figure 5.14: Differences seen when routing help-request packets with 2D-VD (in red) and with 3D-VD (in green).

- dynamically guide servers to ongoing hotspots

Here, we use classical Newtonian gravitational law (effect of mass) within this 3D-VD framework, guiding servers towards any occurring hotspots. The goal is to avoid stationary servers that rigidly serve only pre-designated areas. Instead, they would move around autonomously within the VE, bringing resources to bear on load-critical areas as they develop.

5.5.1 Dissemination with 3D-VD

The first step is to actually know where these hotspots are, and to quickly propagate knowledge of their existence to interested entities (i.e., high capacity servers policing the P2P-VE). This is non-trivial in a system lacking any centralised monitoring apparatus. Moreover, simply broadcasting such events would be a waste of bandwidth in an already limited system. This is where 3D-VD's useful properties can again help, by discriminating between peers and peer types, arranging them in multiple strata within the VE. Factoring in the “bottom heavy” peer distributions in VEs, where constrained nodes typically out-

number high capacity ones, 3D-VD eases the dissemination of packets that warn of critical load areas.

This feature is illustrated in in Figure 5.14, where the Z -axis exposes a direct link between the source and target nodes. In this case, the source is a higher capacity peer (super-node) and has detected that it and its neighbours (NB) are under severe load. As such, it needs to quickly inform the nearest dedicated server (i.e., the target) of this developing hotspot.

Four help-request packets are generated and subsequently sent towards the corners of the virtual map. Using a plain 2D-VD, the example packet seen in the figure is propagated slowly, with multiple intermediary hops (red path). In contrast, 3D-VD provides greater neighbour awareness, facilitating timely response via a direct link between the source and target.

With 3D-VD, higher-capacity peers to remain higher up the Z -axis . This, in turn, allows them something akin to better situational awareness, as 3D-VD forces peers in the sparsely populated higher stratum to have more links to peers in the dense lower strata. Refer to the relatively larger Voronoi areas at the top of the 3D-VD cube in Figure 5.2, as opposed to smaller but more numerous polyhedral areas along the bottom.

It must be noted that only relatively higher capacity peer-types (super-nodes and above) are used as detectors. Burdening each and every peer with detection duties is wasteful and, should a hotspot were to occur, all these peers would likely overwhelm the VE with help-requests.

5.5.2 Simulation Parameters

Server numbers for our simulations were set at ratios of roughly 1:150, where 27 servers avail themselves to 3973 peers (for a total of 4000). By comparison, typical server to client ratios for current First-Person Shooter (FPS) games, which are the most sensitive to latency, range from 1:24 to 1:128. Peer distributions for our simulations were set at 4%

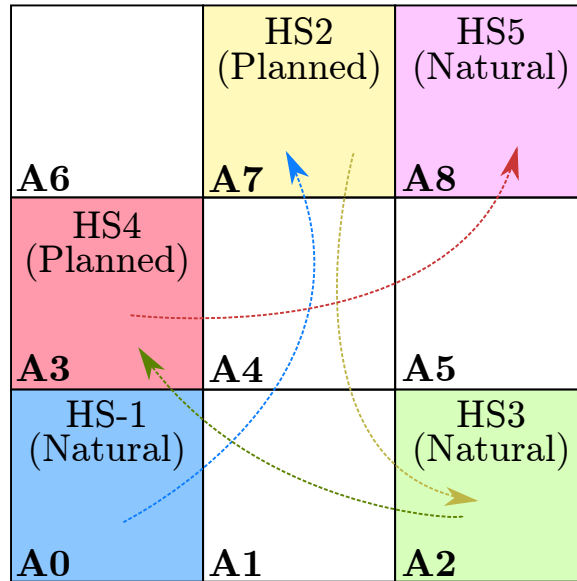


Figure 5.15: Showing the order, placement, dispersal and type of simulated hotspots used in our experimentation.

super-nodes (representing players with higher available network bandwidth), 48% normal nodes (average capacity) and 48% weak-nodes (bandwidth constrained). This reflects the typical bottom-heavy nature of P2P-VEs, with a few servers being seeded into the virtual map to facilitate game-play.

Consequently, two types of hotspots were designed to mimic hotspots and the peer flocking behaviour:

- **Planned Hotspots** - These model pre-planned behaviours, where groups of players can arbitrarily decide to go to a specific location and participate in an event. The numbers involved are quite significant and would likely overload any provisioned resources. Players could be from any starting position and move expeditiously towards their target. This is reflected in our simulations with a random selection of peers that are then placed on a direct approach vector towards the hotspot.
- **Natural Hotspots** - From a player's point of view, events would occur within the

virtual world that appear entirely natural (e.g., in a fantasy game, a dragon suddenly arrives at a nearby town). This constitutes a randomly occurring event that pulls in players within the vicinity (players converge to help defend the town). We model this behaviour by creating a pseudo gravity well, based on Newtonian laws that attracts nearby peers. This technique is explained in the next section.

In Figure 5.15, we imagine looking at the VE from a top-down perspective. This a flattened 2D view of our virtual map, with the standard X and Y axis visible and the Z-axis pointing towards the reader. The map is divided into 9 equal areas and numbered 0 to 8, starting from the bottom left to the top-right. The hotspots are introduced sequentially, alternating between Natural and Planned (NatHS / PlnHS). Their placements were designed to be as far apart as possible, in an effort to fully test the server-attraction qualities in our mechanism. Thus, in our simulations, we have NatHS-1, PlnHS-2, NatHS-3, PlnHS-4 and finally NatHS-5 placed in the order: A0, A7, A2, A3, A8.

It must be said that from a developer's point of view, natural hotspots could be effectively managed and even themselves be used to manage load. Game designers could create

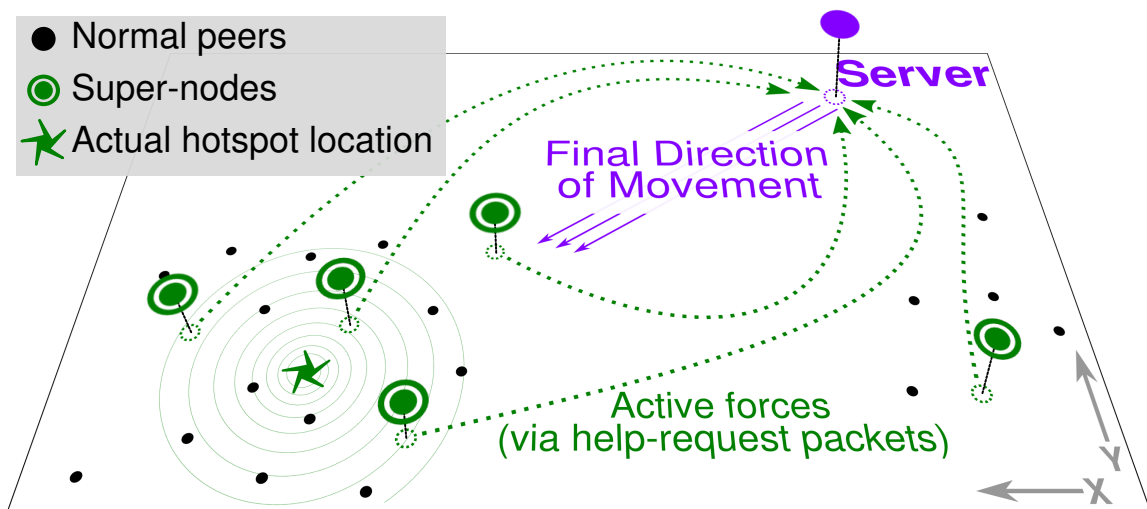


Figure 5.16: Illustrating how servers are pulled towards hotspots.

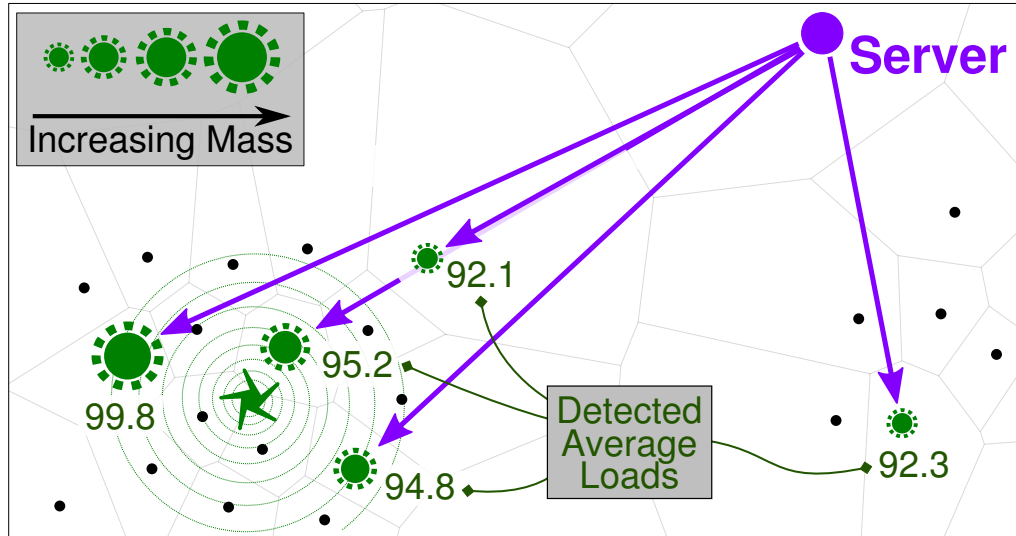


Figure 5.17: Breakdown of force vectors

game-content and events that pull players towards certain zones in the VE, which were earlier earmarked with adequate resources.

However, the issue remains with the estimation of the resources required (2-3 servers? or 20-30?) as it may be difficult to predict the number of players attracted to the event. Moreover, this technique does not address planned hotspots, where players could choose to concentrate in other, unforeseen areas of the virtual world.

5.6 Using Newton's Laws

Considering the three dimensional context, Newton's laws of motion and gravity provide an ideal framework with which to model certain aspects of our approach. Of specific interest is the law of Universal Gravitation, a form of the general inverse-square law widely used in the sciences. The idea is to model how myriad environmental forces can affect a peer within our VE. Specifically, we compute the varied and often competing gravitational forces acting on a server, approximating the effect of being *pulled* towards multiple occurring hotspots.

This mechanism is used to control the movements of these dedicated servers, creating

a directional logic that guides them towards the areas in the virtual map most in need of their resources. Thus, calculating the force, F , between a server and a hotspot, is an adaptation of the well-known formula:

$$F = G \frac{M_C M_H}{D^2} \quad (5.1)$$

where G is the gravitational constant; M_C represents the mass of the computing peer; M_H is the mass of the hotspot in question; and D^2 is the squared distance between the two, emphasizing the weakness of gravity as distances grow larger. Readers will note that this is the same Newtonian equation with which we created our earlier “Mass Effect” selection policies in chapter 3.

Figure 5.16 demonstrates the process of server attraction. We have a perspective view of the virtual map with the Z -axis positions of certain peers exposed by their slight elevations. This represents their often higher arbitration capacities relative to other, more constrained peers (represented by black dots).

The super-nodes (SN) check for signs of stress in their respective areas, which may overlap. They can easily discern their neighbours' loads through their positions on the Z -axis. Once average load crosses a critical threshold, the SNs start sending out help-request packets. These packets represent the gravitational forces exerted on the server by the hotspot. The SNs do not try to estimate the coordinates of the hotspot. Instead, they just report their own locations whenever they detect critical loads. There is little point in trying to ascertain an exact position, as all that is needed is that we bring the servers into the general vicinity of the hotspot.

It is important to note that some SNs request for help even when they are nowhere near a hotspot (see the rightmost SN). This is just the result of temporary load spikes in that area, something akin to mini-hotspots which can be regarded as noise that can potentially “distract” a server from its target. Despite the presence of such noise, the

Table 5.2: Showing typical data within our Help-Request Packets

Name	Type	Description
Time	Int64	The exact time the event is triggered (in simulation ticks)
Vector	[x,y,z]	Coordinates of the reporting super-node
Load	Double	The average network load detected of peers around SN

server is still pulled in the general direction of the hotspot. This is due to the use of force-vector summation. When the server receives a help-request, it determines the vector and distance, D , to the reporting SN, which is an approximation for the hotspot.

For brevity, we do not list the full method of deriving the “hotspot mass”. The term is instead an euphemism for the criticality of the event. Suffice it to say that the higher the average load detected, the higher the mass value (extrapolated to increasingly large values, with a maximum of $1.0e16$). Accordingly, we obtain the gravitational force exerted on the server, which is then added to the uniform vector component to give us a force-vector. Summation of all these received vectors by the server results in a singular direction and magnitude.

Thus, noise is an expected phenomena and is catered for in our approach. It stands to reason that real, active hotspots induce more frenetic peer activity. The resultant load increase is a sustained one, being detected with greater frequency by SNs in the area. This prolonged barrage of help-requests outweighs the distraction of inconsequential mini-hotspots.

5.6.1 Further Experimental Results

For our simulations, a custom engine had to be developed in C++, due to performance concerns and the use of CGAL [Pion and Teillaud, 2009] to derive the requisite 3D-VD. For simplicity, we termed our composite approach **Autonomous Load Management**

(ALM). All simulations were ran with a peer population of 4000, with the distribution of node-types set at 4% Super, 48% Normal and 48% Weak Nodes. Server numbers were restricted to 27, so as to initially put 3 servers within the 9 hotspot areas.

Peers were designed to move randomly, with the exception being if they were under the influence of Hotspots (HS). If, for example, a peer came under the gravitational pull of a Natural HS, it would then have a strong force vector added to its movement. This ensures the peer goes towards the HS, staying within its vicinity and mimicking player behaviour.

The length of each run was capped at 1000 simulation ticks. As mentioned prior, a tick represents a discrete game-play turn and has no appreciable time equivalent. This was in an effort to measure a key metric, the *timeliness* of selecting game-play arbitrators. If a player wished to engage an enemy, the VE must quickly assign a candidate from amongst its NBs.

Thus, the ideal situation is to have an arbitrator assigned at the very next turn. Any rejection (due to already stressed NBs) would be counted as failures, with **Fail-1** being the 1st tick failure, **Fail-2** the 2nd, and so on. Peers would give up after 3 tries (**GaveUp**) or

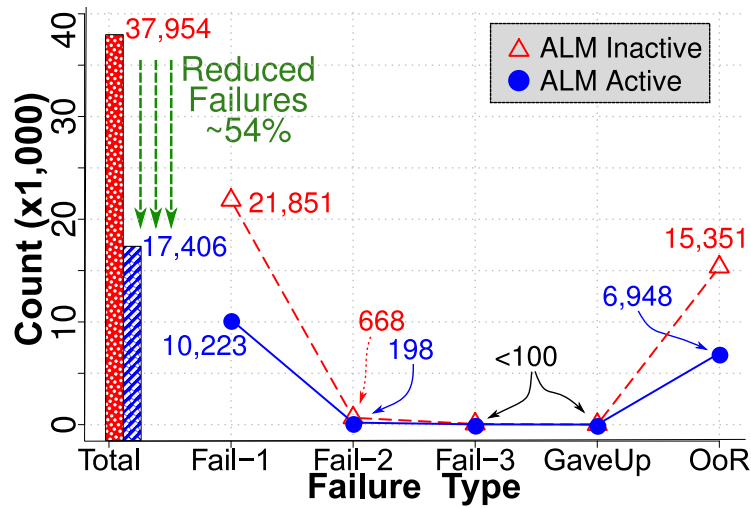


Figure 5.18: Showing decrease in failure rates when activating the extended ALM technique alongside 3D-VD.

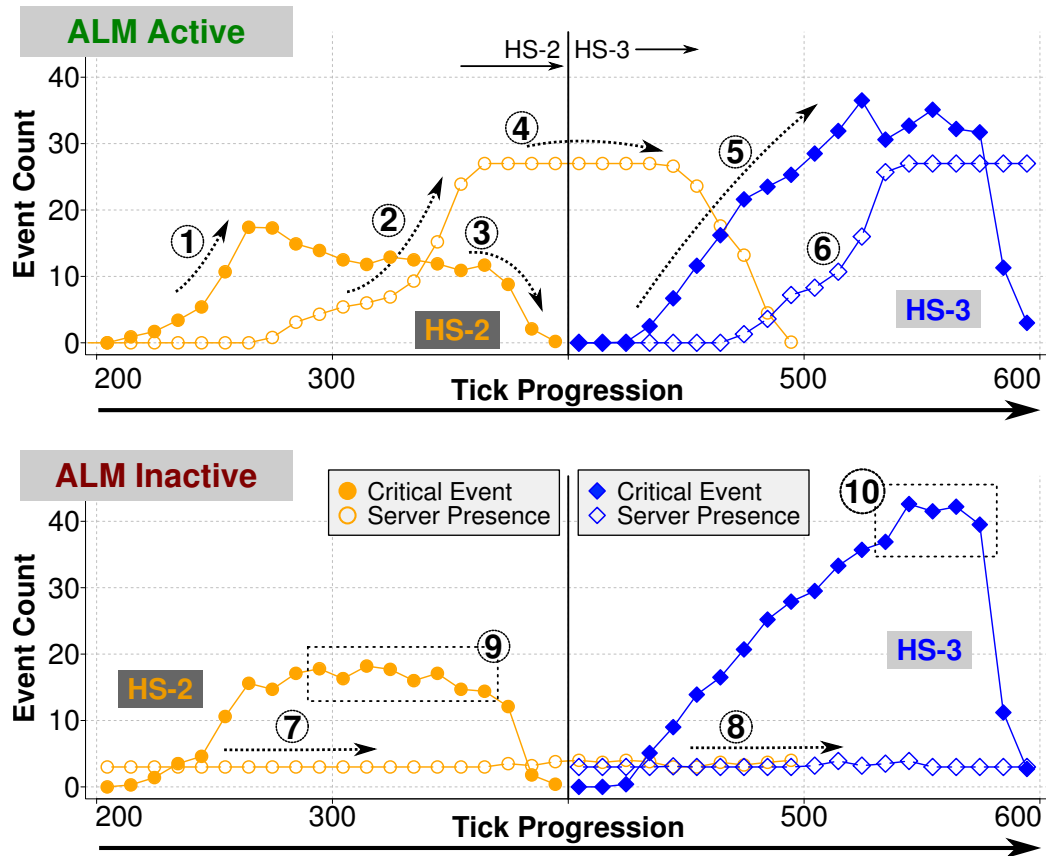


Figure 5.19: Hotspot dynamics with ALM (top) and without (bottom)

if it moves out of range (**OoR**). The onus on ALM was to fluidly provide server resources *when and where* they are needed and thus, maximise their utility.

We start with Figure 5.18, where ALM is shown to reduce failure rates when activated. Total failures were minimized by 54%, greatly improving game-play experience within the VE. There is significant reduction at Fail-1, with a flow-on effect decreasing the chances of subsequent failures. Simply put, if we get it right the first time, the peer will not try again, ultimately lessening the probability of subsequent failures.

We also minimize the impact of subsequent Fail-OoRs, as interacting peers in hotspot areas are more likely to lose connectivity to each other, similar to how one would lose a friend in a crowd. There is significant chance that peers become disconnected (i.e., no

longer NBs) after the first failed tick. Failures still exist, due to the delay from the start of the HS to the eventual arrival of servers to the HS. As ALM is only reactive, loads at the early stages remain entirely unchecked.

In Figure 5.19, the chronological progression of two example hotspots (HS-2 and HS-3) are shown, from simulation ticks 200 to 600. At instance **(1)**, just after the creation of HS-2, we see an initial increase in the detection of Critical Events (CE). Slowly, servers arrive to aid the peers at **(2)** and we see a corresponding drop in CEs at **(3)**. All servers arrive at tick 275, as evidenced by the plateau at **(4)** that overflows onto the initial phase of HS-3. At this stage, HS-3 has only just been generated, and is not actively triggering CEs and thus, we see how the servers continue to linger within the vicinity of HS-2.

At instance **(5)**, we see HS-3 gaining momentum, causing more numerous CEs. It being a natural hotspot, more peers are pulled towards it, with numbers rising beyond 1200 compared to only 800 for planned hotspots. This was an intended design decision, to eventuate a much larger volume of CEs from a larger, more active peer population. Accordingly, servers gradually shift away from the previous hotspot and move towards the very active HS-3, as seen in **(6)**.

When ALM is deactivated (bottom plots in Fig. 5.19), we see how the CEs are triggered in generally the same fashion. However, servers remain stationary within their assigned areas, with no increase in server numbers at HS-2 **(7)**, and again at HS-3 **(8)**. Also, note the higher and more prolonged occurrences of CEs seen for both hotspots at **(9)** and **(10)**. This represents a sure indicator that the reduced presence of servers worsens the load situation, resulting in ever more critical events.

Chapter 6

Conclusion

This chapter draws to a close our novel proposal to apply the geometric constructs of 3D-Voronoi Diagrams onto the domain of fully Peer-to-peer Virtual Environments. Here, we surmise the research aims, the scientific contributions, and the avenues for future work.

6.1 Research Aims

Throughout the work contained herein, the overarching aim has been to enable a fully P2P-VE. The concept at hand is of an immense virtual world, with large volumes of concurrent participants that are all actively engaged in the simulation. The idea itself is not entirely new, and many commercial game developers indeed strive towards such a goal. However, the research challenge here has been the conundrum of enabling such a bandwidth-demanding and time-sensitive application on a *P2P platform*.

In a supposedly decentralised system, there are no central servers to guide and control the connections and services that peers require. Indeed, to enable a P2P-VE, we are quickly faced with the question of how to connect a multitude of players, all with acute heterogeneity in hardware/bandwidth resources. A dynamic and flexible network overlay is needed, with the efficacy of any proposed technique paramount in the quest to maintain consistency and cohesion in the virtual world.

Pursuant to this, a new challenge emerges in the form of arbitration services. In any

interactive environment, the communications and events between peers are the building blocks that drive the virtual world forward. This is doubly important in on-line games, where a competitive tone is the default and players derive enjoyment from combat and collaboration with remote participants. Again, without centralised protocols, how are we to resolve the interactions between peers? The prevailing method of having players connected to a single authoritative server places undue restrictions of virtual world sizes, and more importantly, is decidedly unfit for a P2P architecture.

Even with all the above set in place, the issue then turns to the pursuit of load management, one of the strongest benefits of moving towards P2P architectures and principles. With each peer having a disparate amount of available resources, how do we then manage their capabilities and allow players to both enjoy the VE and, at the same time, contribute towards it in a tangible way? With arbitration services being a key need for any on-line game, how can we leverage the latent capacities of participating peers?

Accordingly, 3D-VD was proposed, as a way to connect and cluster the peer population in an efficient, timely and scalable manner. The novel addition of the 3rd-dimension to basic 2D varieties used in prior art affords more flexibility in managing inter-peer connections and forms the basis for further progress. Part of the work herein details the exploration and verification of various aspects of the 3D-VD technique, with feasibility a major hurdle in any attempts to utilize it in the latency sensitive domain of on-line games. The specific research questions are surmised as follows:

- Knowing the cost of derivation, can we use 3D-VD in the area of P2P-VEs?
- How do we apply 3D-VD to on-line-games, and what augmentations are needed for its use in arbitration services?
- Can we reduce the increased network-bandwidth costs of 3D-VD?
- How do we maximise the load balancing features of 3D-VD in a P2P platform?

6.2 Contributions

The preceding chapters have each addressed of the stated objectives of this thesis. The research contributions here can be surmised into four distinct major areas:

1. Feasibility of 3D-VD

The first step of our approach is to ascertain the feasibility of moving up to a three dimensional VD. The computational cost when deriving such geometric construct is an exceedingly important factor, especially when we take into account the demanding nature of on-line games and VEs in general. With FPS games and their strict network latency demands being a key target area, there is a need to evaluate the timeliness of 3D-VD construction.

Pursuant to this, we have detailed the 3D-VD technique, its strengths and requirements. A custom C++ simulation engine was built for the extensive experimentation to be conducted later, and the question of feasibility was addressed in our earliest simulations, with computational speed coming within the range needed to ensure responsive game-play. With this, we enable a vital component to the sense of immersion players feel within the virtual world. The 3D-VD mechanism shows a dynamism and aptitude to the task, whilst minimizing any negative impacts.

The discussion here also concerns the various aspects of enabling a fully P2P virtual environment. Concepts such as the role of in-game arbitrators, the nature of FPS games, and the general platform for a P2P-VE are explained in detail. This provides the reader with sufficient background and an understanding of the constituent research fields that 3D-VD is built upon.

2. Application of 3D-VD

The topic then turns towards the actual application of the 3D-VD technique, and the various selection policies we have developed to aid in securing arbitration services for the peer population. The 3D-VD method shows great flexibility, where the Z-axis can be adapted to suit various requirements. Our work has concentrated on using the most limiting peer resource, that of upload bandwidth as the main metric. Consequently, the Z-axis reflects a peer's dynamic upload capability.

As for the selection policies, 3D-VD enabled us to discern the spatial relationships and allowed the application of rudimentary physical laws onto a 3D virtual space. The simple FIT policy was shown to be exceptional but contains conceptual flaws that may hinder its use. The mass-effect policies exhibited inadequate results but remain an interesting avenue for further research. Further augments may satisfy the need to balance nearness (for timeliness), fairness (for security) and resource availability (to minimise retries).

The question we posed was that of timely arbitrator selection. That is, in an egalitarian network of peers, (i) who do we select as temporary game-play adjudicators; (ii) how do we select this peer; and (iii) can we select him fast enough to avoid delaying game-play. Such issues were addressed and a discussion on the possible metrics for use as the third dimension was also provided.

3. Bandwidth Reduction

An issue that subsequently arose was the escalation in upload requirements caused by 3D-VD. With more exposure to adjacent neighbours, peers were being forced to send updates at a rate that exceeded standard broadband upload bandwidths. We thus augmented our algorithm with a peer classification mechanism that significantly reduced the bandwidth demands at each peer. Various simulations are performed and their results and accompanying analysis are also provided.

The augmentation involved a two-tiered differentiation model that limited upload rates to neighbours that were logically “further away” from the computing peer. With the flexibility afforded by 3D-VD, we do not rely on simple and static proximity measures (so often used in prior schemes) and instead, use the occlusion information provided by 3D-VD. This is indeed an additional layer of interest management that is more nuanced and subtle in nature.

The results show how we successfully effected reductions in the upload rates across the entire peer population, thus removing the need for more complex schemes and ultimately allowing the utilization of 3D-VD. Crucially, we also fluidly maintained VE consistency, and prevented any disruption to VE cohesion that could be caused by lost or disrupted inter-peer connections.

4. Load Management

Here, the attention moves towards an examination of the load-balancing effects of 3D-VD as they are applied to P2P-VEs. The core mechanism is augmented with innovative techniques that use Newton’s theories on gravity, in order to model certain types of hotspot behaviours. More importantly, the same principles are used to implement a mechanism that guides servers towards these hotspots in order to alleviate load.

3D-VD is again of great use as it adds much needed flexibility for the detection of hotspot occurrences and the subsequent dissemination of help-request packets towards policing servers. A variety of peer configurations was tested, with peer types spread across several peer populations. Mainly using the FIT/FAS arbitrator selection policies from prior chapters, results were compared against a randomised selection mechanism (RAS). Further analysis on the traces for our anti-flocking mechanism was also performed, constituting our idea for an autonomous load-management system.

It is shown that 3D-VD exhibits: (i) excellent load-balancing capabilities; (ii) intelligent load direction to appropriate peer types; and (iii) an ability to mitigate the concerns associated with player-flocking behaviour.

6.3 Future Work

The work presented here is ongoing and represents the initial stages of an exploratory study on the use of 3D Voronoi Diagrams. The application of 3D-VD within the context of P2P games and VE is a nascent area of study. It builds on the fundamentals of using simple two-dimensional VDs, in an effort to address other issues within the realm of traditional field of DVEs. We note the many desirable features it introduces and illustrate various characteristics in the work detailed herein.

Alternative Metrics

Our approach, while novel, need thorough testing against other forms of arbitrator selection. We are continuing to look at the metrics that may be employed and, of particular interest, is the use of *composite metrics* that may provide suitable mechanisms to address the problems in our research context. We also note possible drawbacks and the potential improvements to be made to our approach, such as:

- Malicious users can modify their capacity/peer-load metric so as to avoid becoming arbitrators. This results in unfair allocation of processing.
- The security question: How do we reconcile the use of in-game coordinates to perform peer clustering with the Lock-Step protocol, where peer's in-game coordinates are meant to be hidden from each other?

The above-mentioned issues form interesting research areas, and will hopefully be addressed in the near future. For now, we have used a non-positional metric in the form

of individual peers' upload bandwidth capacities as the unit of measure along the third dimension. This dynamic metric reacts to the movement of the peers within the virtual world and has proven to be an apt choice.

Scale of Simulations

The population size utilized throughout our experimentation may be viewed as inadequate, especially given the stated possibility of a fully-decentralised P2P online-game with potential reach in the order of hundreds of concurrent players. The hardware and computational constraints at the time, and the need for an in-depth exploration of 3D-VD and arbitrator selection policies, restricted our tested populations to 1000 to 2000 nodes.

As the state of GPGPU frameworks has progressed, along with the increasing accessibility of cloud-computation services, it is a reasonable assumption that more strenuous simulations can be performed with actual node numbers of 100,000 and above. This would be in-line with popular MMOGs, some of whom boast concurrent users exceeding this size, albeit in games that are typically slower-paced (non-FPS games), and with varying degrees of virtual world compartmentalization.

Bandwidth Reduction Methods

With regards to the bandwidth reduction aspects of our work, an immediate continuation would be to understand the effects of further reducing the frequency of updates to BounBs. Specifically, what are the negative consequences of having less updates and how will it impact game-play. The aim is to model the Euclidean distances of a peer's neighbours on heavy-tailed probability distributions and thus, vary the rate of updates accordingly.

Such a mechanism will achieve a "smoothing" effect, in contrast to our current two-step classification approach (30 updates/second for enclosing neighbours and then a sudden drop to 1 u/s for boundary neighbours). The aim is to achieve the same bandwidth reductions yet improve the rate of updates without affecting the consistency of the VE.

Load Management Improvements

Further areas for improvement in load-management include making the proposed approach more robust when peer populations are extremely low, and switching the current metric with that of processing capacities instead of purely accounting for the network bandwidth demands. It would also be interesting to investigate the introduction of composite metrics that incorporate bandwidth, processing and storage capacities. Finally, a key concern is to enable a predictive way to guide servers towards potential hotspots, instead of the purely reactive model used.

6.4 Final Words

Enhancing the classical 2D-VDs often used in P2P-VEs for fixed networks, we introduced our unique utilization of a 3rd dimension that represents an individual peer's network upload capacities. This approach is then extended by incorporating an appropriate policy. The Z -axis positioning and the resulting information from 3D-VD is effectively employed to choose the fittest neighbour to adjudicate the interactions between two competing peers.

With this working premise, we show the feasibility and benefits attached to the use of 3D-VD and have addressed several real concerns with the application of P2P principles onto distributed VEs. This work presented has made significant advances in the area of on-line games and VEs in general, and we are perhaps a step closer to enabling a fully P2P-VE.

Bibliography

- L. Achterbosch, R. Pierce, and G. Simmons. Massively multiplayer online role-playing games: the past, present, and future. *Comput. Entertain.*, 5(4):1–33, 2007. doi: 10.1145/1324198.1324207. Cited on pages 3 and 24.
- D. T. Ahmed and S. Shirmohammadi. Uniform and {Non-Uniform} Zoning for Load Balancing in Virtual Environments. In *2010 5th International Conference on Embedded and Multimedia Computing {(EMC)}*, pages 1–6. IEEE, Aug. 2010. ISBN 978-1-4244-7710-4. doi: 10.1109/EMC.2010.5575761. Cited on page 98.
- M. Albano, L. Ricci, and L. Genovali. Hierarchical p2p overlays for DVE: An Additively Weighted Voronoi based approach. In *Ultra Modern Telecommunications Workshops, 2009. ICUMT '09. International Conference on*, pages 1–8, 2009. doi: 10.1109/ICUMT.2009.5345362. Cited on page 97.
- M. Almashor and I. Khalil. Reducing network load in large-scale, peer-to-peer virtual environments with 3d voronoi diagrams. In *High Performance Computing, 2010. HiPC 2010. 17th Annual IEEE International Conference on*, 2010a. Cited on pages 24 and 116.
- M. Almashor and I. Khalil. Load-balancing properties of 3d voronoi diagrams in peer-to-peer virtual environments. In *Parallel and Distributed Systems (ICPADS), 2010 IEEE 16th International Conference on*, 2010b. doi: 10.1109/ICPADS.2010.97. Cited on pages xvi, 97, and 100.

- M. Almashor, I. Khalil, and G. Leach. Dynamic game-play arbitrators with 3d voronoi diagrams. In *Network Computing and Applications, 2010. NCA 2010. Ninth IEEE International Symposium on*, 2010. doi: 10.1109/NCA.2010.46. Cited on pages 7 and 24.
- G. Armitage. An experimental estimation of latency sensitivity in multiplayer quake 3. In *Networks, 2003. ICON2003. The 11th IEEE International Conference on*, pages 137–141, 2003. ISBN 1531-2216. doi: 10.1109/ICON.2003.1266180. Cited on pages 27, 28, 29, 38, and 66.
- L. B. Baker. Factbox: A look at the \$65 billion video games industry. Online (Accessed 2011-11-16), June 2011. URL <http://uk.reuters.com/article/2011/06/06/us-videogames-factbox-idUKTRE75552I20110606>. <http://uk.reuters.com/article/2011/06/06/us-videogames-factbox-idUKTRE75552I20110606>. Cited on page 24.
- N. E. Baughman, M. Liberatore, and B. N. Levine. Cheat-Proof payout for centralized and Peer-to-Peer gaming. *Networking, IEEE/ACM Transactions on*, 15(1):1–13, 2007. ISSN 1063-6692. doi: 10.1109/TNET.2006.886289. Cited on pages 8, 27, 33, 37, 53, 66, 67, 68, and 95.
- Y. W. Bernier. Latency compensating methods in Client/Server in-game protocol design and optimization. In *Communications Proceedings of the 15th Games Developers Conference*, 2001. URL <http://developer.valvesoftware.com/wiki/>. Cited on pages 30, 34, 37, 66, 68, and 96.
- C. E. Bezerra, F. R. Cecin, and C. F. R. Geyer. A3: A Novel Interest Management Algorithm for Distributed Simulations of {MMOGs}. pages 35–42, 2008. ISBN 1550-6525. doi: 10.1109/DS-RT.2008.11. Cited on page 97.
- A. Bharambe, J. R. Douceur, J. R. Lorch, T. Moscibroda, J. Pang, S. Seshan, and X. Zhuang. Donnybrook: enabling large-scale, high-speed, peer-to-peer games. In *SIG-*

- COMM '08: Proceedings of the ACM SIGCOMM 2008 conference on Data communication*, pages 389–400, New York, NY, USA, 2008. ACM. ISBN 978-1-60558-175-0. doi: <http://doi.acm.org/10.1145/1402958.1403002>. Cited on pages 6, 23, 24, 48, 54, 67, 68, 69, 78, 81, 82, 96, 97, and 101.
- A. Boukerche, A. Zarrad, and R. Araujo. A Cross-Layer Approach-Based gnutella for collaborative virtual environments over mobile ad hoc networks. *Parallel and Distributed Systems, IEEE Transactions on*, 21(7):911–924, 2010. ISSN 1045-9219. doi: 10.1109/TPDS.2009.91. Cited on pages 3 and 25.
- E. Buyukkaya, M. Abdallah, and R. Cavagna. VoroGame: a hybrid P2P architecture for massively multiplayer games. In *Consumer Communications and Networking Conference, 2009. 6th IEEE*, 2009. doi: 10.1109/CCNC.2009.4784788. Cited on pages 34, 36, 37, 66, 67, 68, 69, and 97.
- J. Chen, B. Wu, M. Delap, B. Knutsson, H. Lu, and C. Amza. Locality aware dynamic load management for massively multiplayer games. In *Proceedings of the tenth {ACM} {SIGPLAN} symposium on Principles and practice of parallel programming*, pages 289–300, Chicago, {IL,} {USA}, 2005. ACM. ISBN 1-59593-080-9. doi: 10.1145/1065944.1065982. Cited on page 98.
- K. Chen, P. Huang, and C. Lei. Effect of network quality on player departure behavior in online games. *Parallel and Distributed Systems, IEEE Transactions on*, 20(5):593–606, 2009. ISSN 1045-9219. doi: 10.1109/TPDS.2008.148. Cited on pages 6, 21, 27, 28, 29, 38, 52, 66, and 92.
- M. Claypool. The effect of latency on user performance in real-time strategy games. *Computer Networks*, 49(1):52 – 70, 2005. ISSN 1389-1286. doi: 10.1016/j.comnet.2005.04.008. URL <http://www.sciencedirect.com/science/article/pii/S1389128605001003>. jce:titlejNetworking Issue in Entertainment Computingi/ce:titlej. Cited on page 29.

- M. Claypool and K. Claypool. Latency and player actions in online games. *Commun. ACM*, 49(11):40–45, Nov. 2006. ISSN 0001-0782. doi: 10.1145/1167838.1167860. URL <http://doi.acm.org/10.1145/1167838.1167860>. Cited on pages 2, 3, 5, 24, 29, and 66.
- A. Corman, S. Douglas, P. Schachte, and V. Teague. A secure event agreement (SEA) protocol for peer-to-peer games. In *Availability, Reliability and Security, 2006. ARES 2006. The First International Conference on*, page 8 pp., 2006. doi: 10.1109/ARES.2006.15. Cited on pages 8, 27, 33, 37, 67, 68, and 96.
- A. Denault, C. Canas, J. Kienzle, and B. Kemme. Triangle-based obstacle-aware load balancing for massively multiplayer games. In *Network and Systems Support for Games (NetGames), 2011 10th Annual Workshop on*, pages 1–6, 2011. doi: 10.1109/NetGames.2011.6080980. Cited on page 97.
- ESA. Essential facts about the computer and video game industry 2011. Online (Accessed 2011-11-16), 2011. URL http://www.theesa.com/facts/pdfs/ESA_EF_2011.pdf. http://www.theesa.com/facts/pdfs/ESA_EF_2011.pdf. Cited on page 24.
- C. GauthierDickey, V. Lo, and D. Zappala. Using n-trees for scalable event ordering in peer-to-peer games. In *NOSSDAV '05: Proceedings of the international workshop on Network and operating systems support for digital audio and video*, pages 87–92, New York, NY, USA, 2005. ACM. ISBN 1-58113-987-X. doi: <http://doi.acm.org/10.1145/1065983.1066005>. Cited on page 97.
- J. Goodman and C. Verbrugge. A peer auditing scheme for cheat elimination in MMOGs. In *Proceedings of the 7th ACM SIGCOMM Workshop on Network and System Support for Games*, pages 9–14, Worcester, Massachusetts, 2008. ACM. ISBN 978-1-60558-132-3. Cited on pages 8, 27, 33, 67, and 68.

- L. Guo, S. Chen, Z. Xiao, E. Tan, X. Ding, and X. Zhang. A performance study of {BitTorrent-like} peer-to-peer systems. *Selected Areas in Communications, {IEEE} Journal on*, 25(1):155–169, 2007. ISSN 0733-8716. doi: 10.1109/JSAC.2007.070116. Cited on page 28.
- S. Hu, J. Chen, and T. Chen. VON: a scalable peer-to-peer network for virtual environments. *Network, IEEE*, 20(4):22–31, 2006. ISSN 0890-8044. doi: 10.1109/MNET.2006.1668400. Cited on pages xiv, 2, 3, 6, 21, 23, 26, 34, 35, 36, 37, 38, 42, 44, 46, 47, 48, 49, 53, 60, 67, 68, 69, 72, 78, 82, 94, and 100.
- S. Hu, S. Chang, and J. Jiang. Voronoi state management for Peer-to-Peer massively multiplayer online games. In *Consumer Communications and Networking Conference, 2008. CCNC 2008. 5th IEEE*, pages 1134–1138, 2008. ISBN 0197-2618. doi: 10.1109/ccnc08.2007.255. Cited on pages 35, 36, 37, and 97.
- S.-Y. Hu and K.-T. Chen. Vso: Self-organizing spatial publish subscribe. In *Self-Adaptive and Self-Organizing Systems (SASO), 2011 Fifth IEEE International Conference on*, pages 21–30, oct. 2011. doi: 10.1109/SASO.2011.13. Cited on pages 67, 68, 70, and 97.
- P. Kabus and A. P. Buchmann. Design of a cheat-resistant {P2P} online gaming system. In *Proceedings of the 2nd international conference on Digital interactive media in entertainment and arts*, pages 113–120, Perth, Australia, 2007. ACM. ISBN 978-1-59593-708-7. doi: 10.1145/1306813.1306840. Cited on page 96.
- J. Keller and G. Simon. Toward a peer-to-peer shared virtual reality. In *Distributed Computing Systems Workshops, 2002. Proceedings. 22nd International Conference on*, pages 695–700, 2002. doi: 10.1109/ICDCSW.2002.1030849. Cited on pages 6, 82, and 97.

- B. Knutsson, H. Lu, W. Xu, and B. Hopkins. Peer-to-peer support for massively multiplayer games. In *INFOCOM 2004. Twenty-third Annual Joint Conference of the IEEE Computer and Communications Societies*, volume 1, page 107, 2004. ISBN 0743-166X. doi: 10.1109/INFCOM.2004.1354485. Cited on pages 5, 21, 27, 29, 66, 92, and 96.
- S. Krause. Coping with hotspots: {AoI} adaption strategies for {P2P} Networked Virtual Environments. In *International Conference on Ultra Modern Telecommunications & Workshops, 2009. {ICUMT} '09*, pages 1–6. IEEE, Oct. 2009. ISBN 978-1-4244-3942-3. doi: 10.1109/ICUMT.2009.5345511. Cited on page 98.
- D. Kushner. Engineering EverQuest: online gaming demands heavyweight data centers. *Spectrum, IEEE*, 42(7):34–39, 2005. ISSN 0018-9235. doi: 10.1109/MSPEC.2005.1460347. Cited on pages 3, 5, and 21.
- D. Kushner. Betting the farm on games. *Spectrum, IEEE*, 48(6):70–88, 2011. ISSN 0018-9235. doi: 10.1109/MSPEC.2011.5783711. Cited on page 24.
- M. Kwok and J. Wong. Scalability analysis of the hierarchical architecture for distributed virtual environments. *Parallel and Distributed Systems, IEEE Transactions on*, 19(3):408–417, 2008. ISSN 1045-9219. doi: 10.1109/TPDS.2007.70730. Cited on pages 3, 24, and 27.
- S. Legtchenko, S. Monnet, and G. Thomas. Blue banana: resilience to avatar mobility in distributed mmogs. In *Dependable Systems and Networks (DSN), 2010 IEEE/IFIP International Conference on*, pages 171–180, 28 2010-july 1 2010. doi: 10.1109/DSN.2010.5544919. Cited on pages 67 and 68.
- H. Liu and Y. Lo. DaCAP - a distributed Anti-Cheating peer to peer architecture for massive multiplayer on-line role playing game. In *Cluster Computing and the Grid*,

2008. *CCGRID '08. 8th IEEE International Symposium on*, pages 584–589, 2008. doi: 10.1109/CCGRID.2008.49. Cited on pages 22, 27, 67, and 68.
- M. Macedonia. Games soldiers play. *Spectrum, IEEE*, 39(3):32–37, 2002. ISSN 0018-9235. doi: 10.1109/6.988702. URL 10.1109/6.988702. Cited on pages 3 and 25.
- M. Macedonia. Generation 3D: living in virtual worlds. *Computer*, 40(10):99–101, 2007. ISSN 0018-9162. doi: 10.1109/MC.2007.348. Cited on pages 3 and 25.
- M. Macedonia and M. Zyda. A taxonomy for networked virtual environments. *Multimedia, IEEE*, 4(1):48–56, 1997. ISSN 1070-986X. doi: 10.1109/93.580395. Cited on pages 3 and 24.
- M. Matijasevic, D. Gracanin, K. P. Valavanis, and I. Lovrek. A framework for multiuser distributed virtual environments. *Systems, Man, and Cybernetics, Part B: Cybernetics, {IEEE} Transactions on*, 32(4):416–429, 2002. ISSN 1083-4419. doi: 10.1109/TSMCB.2002.1018762. Cited on page 24.
- A. McCoy, D. Delaney, and T. Ward. Game-state fidelity across distributed interactive games. *Crossroads*, 12(1):3–3, 2005. doi: 10.1145/1144382.1144385. Cited on pages 6, 51, 54, 66, 92, and 96.
- M. Merabti and A. E. Rhalibi. Peer-to-peer architecture and protocol for a massively multiplayer online game. In *Global Telecommunications Conference Workshops, 2004. {GlobeCom} Workshops 2004. {IEEE}*, 2004. doi: 10.1109/GLOCOMW.2004.1417631. Cited on pages 27 and 97.
- P. R. Messinger, E. Stroulia, K. Lyons, M. Bone, R. H. Niu, K. Smirnov, and S. Perelgut. Virtual worlds – past, present, and future: New directions in social computing. *Decision Support Systems*, 47(3):204–228, June 2009. ISSN 0167-9236. doi: 10.1016/j.dss.2009.02.014. Cited on pages 3 and 25.

- P. Morillo, S. Rueda, J. Orduna, and J. Duato. A Latency-Aware partitioning method for distributed virtual environment systems. *Parallel and Distributed Systems, IEEE Transactions on*, 18(9):1215–1226, 2007. ISSN 1045-9219. doi: 10.1109/TPDS.2007.1055. Cited on page 21.
- C. Neumann, N. Prigent, M. Varvello, and K. Suh. Challenges in peer-to-peer gaming. *{SIGCOMM} Computer Communication Review*, 37(1):79–82, 2007. doi: 10.1145/1198255.1198269. URL <http://portal.acm.org/citation.cfm?id=1198269>. Cited on pages 27 and 95.
- S. Pion and M. Teillaud. 3d triangulations. In CGAL Editorial Board, editor, *CGAL User and Reference Manual*. CGAL Editorial Board, 3.5 edition, 2009. Cited on pages 39, 42, 83, 105, and 127.
- M. Preuss, N. Beume, H. Danielsiek, T. Hein, B. Naujoks, N. Piatkowski, R. Stuer, A. Thom, and S. Wessing. Towards intelligent team composition and maneuvering in Real-Time strategy games. *IEEE Transactions on Computational Intelligence and AI in Games*, 2(2):82–98, June 2010. ISSN 1943-068X. doi: 10.1109/TCIAIG.2010.2047645. Cited on pages 98 and 119.
- P. Quax, P. Monsieurs, W. Lamotte, D. De Vleeschauwer, and N. Degrande. Objective and subjective evaluation of the influence of small amounts of delay and jitter on a recent first person shooter game. In *NetGames '04: Proceedings of 3rd ACM SIGCOMM workshop on Network and system support for games*, pages 152–156, New York, NY, USA, 2004. ACM. ISBN 1-58113-942-X. doi: <http://doi.acm.org/10.1145/1016540.1016557>. Cited on page 29.
- A. E. Rhalibi, M. Merabti, and Y. Shen. {AoIM} in peer-to-peer multiplayer online games. In *Proceedings of the 2006 {ACM} {SIGCHI} international conference on Advances in*

- computer entertainment technology*, page 71, Hollywood, California, 2006. ACM. ISBN 1-59593-380-8. doi: 10.1145/1178823.1178907. URL <http://portal.acm.org/citation.cfm?id=1178907>. Cited on page 27.
- S. Rhee, R. Ziegler, J. Park, M. Naef, M. Gross, and M. Kim. Low-Cost telepresence for collaborative virtual environments. *Visualization and Computer Graphics, IEEE Transactions on*, 13(1):156–166, 2007. ISSN 1077-2626. doi: 10.1109/TVCG.2007.17. Cited on pages 3 and 25.
- S. Rooney, D. Bauer, and R. Deydier. A federated peer-to-peer network game architecture. *Communications Magazine, IEEE*, 42(5):114–122, 2004. ISSN 0163-6804. doi: 10.1109/MCOM.2004.1299353. Cited on pages 6, 22, 66, and 68.
- P. Ross. Cloud computing’s killer app: Gaming. *Spectrum, IEEE*, 46(3):14, 2009. ISSN 0018-9235. doi: 10.1109/MSPEC.2009.4795441. Cited on pages 3, 24, and 93.
- A. I. T. Rowstron and P. Druschel. Pastry: Scalable, Decentralized Object Location, and Routing for {Large-Scale} {Peer-to-Peer} Systems. In *Proceedings of the {IFIP/ACM} International Conference on Distributed Systems Platforms Heidelberg*, pages 329–350. Springer-Verlag, 2001. ISBN 3-540-42800-3. Cited on page 28.
- S. Rueda, P. Morillo, and J. M. Ordua. A comparative study of awareness methods for peer-to-peer distributed virtual environments. *Computer Animation and Virtual Worlds*, 19(5):537–552, 2008. ISSN 15464261. doi: 10.1002/cav.230. Cited on pages 3, 26, 60, and 94.
- F. Safaei, P. Boustead, C. Nguyen, J. Brun, and M. Dowlatshahi. Latency-driven distribution: infrastructure needs of participatory entertainment applications. *Communications Magazine, IEEE*, 43(5):106–112, May 2005. ISSN 0163-6804. doi: 10.1109/MCOM.2005.1453430. Cited on pages 6 and 56.

- J. R. Shewchuk. Lecture notes on delaunay mesh generation. Online (CiteSeerX), 1999. URL <http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.102.3695>. Cited on pages 34, 35, 42, 46, 72, and 93.
- J. R. Shewchuk. Delaunay refinement algorithms for triangular mesh generation. *Computational Geometry*, 22(1-3):21–74, May 2002. ISSN 0925-7721. doi: 10.1016/S0925-7721(01)00047-5. Cited on pages 24 and 46.
- M. Skibinsky. The quest for holy scale part 1 : Large-Scale computing. In *Massively Multiplayer Game Development 2*, pages 339–354. Charles River Media, 2005. ISBN 1-58450-390-4. Cited on pages 5, 6, 21, 27, 66, and 68.
- A. Steed and C. Angus. Supporting scalable peer to peer virtual environments using frontier sets. pages 27–34, 2005. ISBN 1. doi: 10.1109/VR.2005.1492750. Cited on pages 6, 27, and 97.
- A. Steed and M. F. Oliveira. Chapter 10 - requirements. In *Networked Graphics*, pages 313 – 353. Morgan Kaufmann, Boston, 2010. ISBN 978-0-12-374423-4. doi: 10.1016/B978-0-12-374423-4.00010-0. URL <http://www.sciencedirect.com/science/article/pii/B9780123744234000100>. Cited on pages 2 and 66.
- M. Steiner and E. Biersack. Ddc: A dynamic and distributed clustering algorithm for networked virtual environments based on p2p networks. In *INFOCOM 2006. 25th IEEE International Conference on Computer Communications. Proceedings*, pages 1 –6, 23-29 2006. doi: 10.1109/INFOCOM.2006.47. Cited on pages 47 and 97.
- T. Sweeney. Unreal networking architecture, 1999. URL <http://unreal.epicgames.com/Network.htm>. Cited on pages 30, 34, 66, 68, and 96.
- M. Varvello, E. Biersack, and C. Diot. Dynamic clustering in delaunay-based P2P networked virtual environments. In *NetGames '07: Proceedings of the 6th ACM SIGCOMM*

- workshop on Network and system support for games*, pages 105–110, New York, NY, USA, 2007. ACM. ISBN 978-0-9804460-0-5. doi: <http://doi.acm.org/10.1145/1326257.1326276>. Cited on page 97.
- S. D. Webb, S. Soh, and J. Trahan. Secure referee selection for fair and responsive Peer-to-Peer gaming. In *Principles of Advanced and Distributed Simulation, 2008. PADS '08. 22nd Workshop on*, pages 63–71. IEEE Computer Society, 2008. ISBN 978-0-7695-3159-5. Cited on pages 8, 27, 33, 37, 67, 68, and 96.
- A. P. Yu and S. T. Vuong. Mopar: a mobile peer-to-peer overlay architecture for interest management of massively multiplayer online games. In *Proceedings of the international workshop on Network and operating systems support for digital audio and video, NOSS-DAV '05*, pages 99–104, New York, NY, USA, 2005. ACM. ISBN 1-58113-987-X. doi: [10.1145/1065983.1066007](http://doi.acm.org/10.1145/1065983.1066007). URL <http://doi.acm.org/10.1145/1065983.1066007>. Cited on pages 26, 67, 68, and 69.
- Q. Zhou, C. Miller, and V. Bassilious. First person shooter multiplayer game traffic analysis. In *Object Oriented Real-Time Distributed Computing (ISORC), 2008 11th IEEE International Symposium on*, pages 195 –200, 5-7 2008. doi: [10.1109/ISORC.2008.28](http://doi.acm.org/10.1109/ISORC.2008.28). Cited on page 6.
- M. Zyda, J. Hiles, A. Mayberry, C. Wardynski, M. Capps, B. Osborn, R. Shilling, M. Robaszewski, and M. Davis. Entertainment R&D for defense. *Computer Graphics and Applications, IEEE*, 23(1):28–36, 2003. ISSN 0272-1716. doi: [10.1109/MCG.2003.1159611](http://doi.acm.org/10.1109/MCG.2003.1159611). URL [10.1109/MCG.2003.1159611](http://doi.acm.org/10.1109/MCG.2003.1159611). Cited on pages 3 and 25.