支援大型多人線上遊戲伺服器叢集之基於地域性的負載分配策略

李政翰　夏啟賢　黃俊鈞　姜美玲　陳弘勳
國立暨南國際大學 資訊管理研究所
{s99213505, s972130383, s97213068, joanna, s101213508}@ncnu.edu.tw

摘要

大型多人線上遊戲對於伺服器的運算要求日漸增加，使用多台伺服器來進行分散式處理已成為趨勢。我們將原本屬於單一伺服器架構的多人線上遊戲 Stendhal 遊戲變成多台伺服器(Multi-server)的遊戲架構，並且實作在 LVS-CAD 叢集式平台上。而如何分配遊戲地圖及玩家至各伺服器的分配策略影響伺服器間的負載平衡和整體遊戲伺服器叢集的效能，是研究上的一大重點。

於本論文中，我們提出了基於地域性的負載分配機制，在分配地圖至伺服器時，考量了遊戲地圖的資訊與伺服器的負載情形，盡可能由目前負載最輕且與該遊戲地圖具有地域性的伺服器來服務，讓遊戲的負載能夠有效地分散至各個伺服器。我們事先將地圖依地圖的資訊分配到 Game Server 上，系統在遊戲進行後會再依玩家分佈的情形動態地調整，能夠使 Game Server 處理的大都是相鄰的地圖，使地圖盡可能地保持地域的完整性。實驗結果顯示，我們的策略不僅降低整體遊戲伺服器叢集的平均 CPU 使用率，系統效能亦大幅地提升。

關鍵詞：多人線上遊戲、伺服器叢集、內容感知需求分配、負載平衡、遊戲地圖。

1. 前言

大型多人線上遊戲(MMOGs, Massively Multiplayer Online Games)是能讓數千個到數十萬個玩家同一時間進行連線的線上遊戲，在相同的虛擬遊戲世界中進行遊戲。現今遊戲伺服器的網路架構以目前主流的多人線上遊戲來說，是以 Client/Server、Hybrid Client/Server 與 Cluster-based Client/Server 這三種架構為主，Peer-to-Peer 架構是集中在單機遊戲中的多人連線衍生出來。

通常遊戲業者會將虛擬的遊戲世界做切割，讓玩家被分散到不同的伺服器。但當某個區域湧進大量玩家時，會使得伺服器負載過重產生熱點，影響遊戲品質及整體系統的穩定度。將過載伺服器中玩家比較少的小區域轉移到其他伺服器，藉由此方法讓伺服器負載降低，減少熱點形成所產生的影響。另外常見的分散遊戲負載的做法是利用遊戲副本的方式，像是某個獨立的遊戲任務，或者是回合制遊戲(Round Game)，讓玩家能夠在該區域活動。

在本論文中我們採用了 Open-Source 的遊戲 Stendhal [1]，它是一種利用地圖來將玩家群聚的遊戲類型，玩家可以透過傳送點的方式轉換到不同的地圖去進行遊戲。Stendhal 原先是單一伺服器架構的多人線上遊戲，在論文[2]中設計了 Kernel-Level 的 Game Connection Handoff 機制，在儘量不修改遊戲軟體的原則下，讓進行中的遊戲連線可以在不同的伺服器間移轉，使 Stendhal 遊戲變成多台伺服器架構的遊戲，且實作在 Linux Virtual Server with Content-Aware Dispatching (LVS-CAD) [3]叢集式平台上。在多伺服器架構下，設計具負載平衡的需求分配策略(Request Distribution Policy)非常重要。

我們在論文[4]中提出了 Game-Server Locality-Aware Request Dispatching(GSLARD)動態的負載分配機制，讓相鄰的地圖盡可能交由同一台伺服器來服務，這樣不僅可以減少遊戲連線移轉的次數，也減少對遊戲資料庫的存取。於本論文中，我們進一步地提出結合動靜態分配策略優點的混合策略，事先將地圖依地圖的資訊分配到適當的 Game Server 上，遊戲進行後再依玩家分佈的情形動態地調整，實驗結果中顯示，我們所提的混合分配策略不僅有效地降低了整體遊戲伺服器叢集的平均 CPU 使用率，也使伺服器所服務的地圖盡可能地保持完整性，減少 Game Server 都處理不相鄰的地圖，降低地圖破碎性的發生。

2. 相關研究與技術

2.1 Linux Virtual Server with Content-Aware Dispatching (LVS-CAD)

Linux Virtual Server [5]簡稱 LVS，是一個具有高延展性、高可用性及高可靠性等優點的伺服器叢集，由前端伺服器(Front-end Server)或稱為負載平衡器(Load Balancer)，以及後端伺服器(Back-end Server)所組成。而 LVS-CAD 是基於 LVS 實作，但卻是可以依照 Request 內容來分配 Request 的伺服器叢集系統，支援 Content-Aware 的分配機制。當 Client 成功建立連線後，會送出 Request 封包給 Front-end Server 去接收，而 Front-end Server 接收到此 Request 封包時，會透過排程演算法選出適合的 Back-end Server，且將此 Request 封包轉送給所挑選的 Back-end Server 來服務處理。LVS-CAD 平台實作了 TCP Rebuilding 機制，當 Back-end Server 收到此 Request 封包時，會開始進行 TCP 連線的重建，讓

原本 Client 與 Front-end Server 的連線轉移到 Back-end Server 上，讓 Back-end Server 能處理 Client 所傳送的 Request，並且回傳處理後的結果給 Client。其 LVS-CAD 的封包轉送流程圖如圖 1 所示。

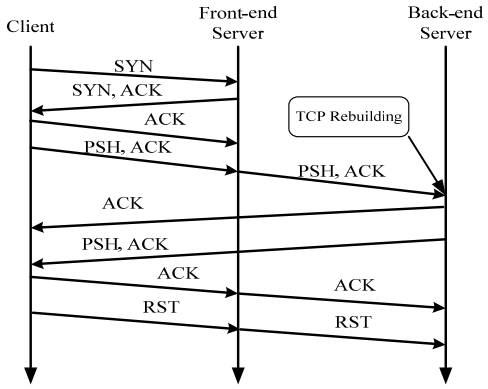

圖 1 LVS-CAD 的封包轉送流程圖

2.2 Game-based Locality-Aware Request Distribution (GLARD) 的動態分配策略

GLARD 負載分配策略是將相同地圖上的玩家分配給同一伺服器來服務。實作中透過 GLARD Hash Table 來記錄每個 Back-end Server 所負責的地圖資訊。當玩家到達另一張地圖時，Game Server 會送出 S2C_TRASFER_REQ 封包訊息給 Client 通知地圖的轉換，而在我們的 LVS-CAD Game Cluster 中，Back-end Server 會將此 S2C_TRASFER_REQ 封包訊息先轉送給 Front-end Server，收到訊息後 Front-end Server 會查詢 GLARD Hash Table 來尋找該地圖是由哪一台 Back-end Server 所負責的。若非原伺服器負責，則會啟動 Game Connection Handoff 機制，將玩家的遊戲狀態轉移給負責該地圖的伺服器來服務，避免玩家們處在同一張地圖，卻分處於不同伺服器上，無法得知玩家彼此之間的訊息，導致遊戲不一致。

2.3 Game-Server Locality Aware Request Distribution (GSLARD)的動態分配策略

GSLARD 負載分配策略主要分為三部分: 一是提出 Locality-Aware Weighted Least Connection(LWLC)來分配地圖，第二是將沒有玩家的地圖重新動態分配，第三是設定地圖的權重，減少各伺服器所分配地圖的權重分配不均所造成的負載不平衡。

LWLC 機制

由於 GLARD 策略是採用 Weighted Least Connection(WLC) 來分配地圖到負載最輕的 Back-end Server，而地圖上的玩家也會被分配到該 Back-end Server。因為 WLC 的分配是只有考慮當時的負載情形，所以 Back-end Server 所負責的地圖有可能是沒有相鄰性的。因此，LWLC 分配機制是希望當地圖被分配時，能先由與該地圖擁有相鄰性且負載最輕的 Back-end Server 來優先分配，若所有相鄰的地圖所處的 Back-end Server 的負載都處於滿載的狀態，則會找尋負載最小的 Back-end Server 做分配。如此一來，具有地域性相鄰的地圖盡可能交由同一台伺服器來服務，這樣可以減少遊戲連線移轉的機會。LWLC 演算法的另一個優點是利用 Server Affinity 的特性，會記錄該玩家的資料在記憶體中，若該玩家再次來到 Back-end Server 所負責的地圖時，玩家資訊直接從該 Back-end Server 記憶體中讀取，減少存取 Database 所造成的 Overhead。

將沒有玩家的地圖重新動態分配

此策略增加了一個動態重新分配地圖的機制，把沒有玩家的地圖從所負責的 Back-end Server 中釋放，等待下次有玩家進入此地圖時，再使用 LWLC 演算法重新分配，如圖 2 所示。是希望讓已經過載的 Server，不需再提供服務給冷門地圖上的玩家，避免負載更嚴重。而等待重新分配的地圖，則可以重新被分配到負載較輕的 Server，使各個 Back-end Server 的資源能充分被利用且負載均衡。

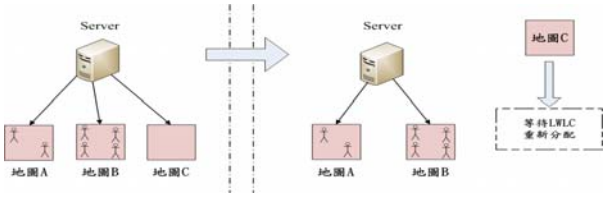

圖 2 動態釋放地圖的示意圖

設定地圖權重，平均分配地圖權重

由於沒有玩家的地圖會被動態地自伺服器釋放，但是若當玩家只是經過某張地圖，則會造成該張地圖都一直被重新分配且再被釋放，使整體系統增加額外的負載。針對此問題，將每張地圖都設立權重值，而權重值的設立是由該地圖所具有的相鄰地圖數量而訂，而設立地圖權重值的另一個優點，則是避免同一台 Back-end Server 服務太多張地圖，故此策略先針對所有遊戲地圖的相鄰情形，對 Back-end Server 設定服務權重值上限，若 Back-end Server 所服務的地圖權重總數，高於或等於所設定的服務權重上限值時，當有地圖要分配到 Back-end Server 時，則該張地圖不會被分配至到達上限值的 Back-end Server。

地圖權重值的分配示意圖如圖 3 所示，圖中兩台 Server 分別服務的地圖張數為 2 與 3，由於每張地圖的權重都不一樣，所以兩台 Server 的服務地圖權重值都相同，但地圖張數卻不同，在設計上再搭配可服務的權重值上限，可以避免 Server 服務太多權重值較重的地圖，讓玩家人數分流且各 Server 的負載能夠較均衡。

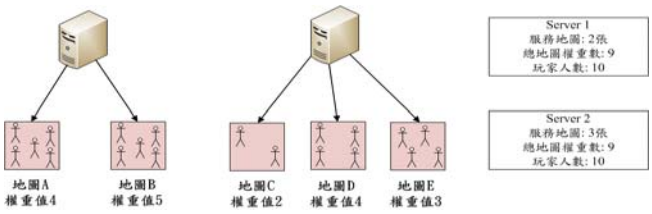

圖 3 地圖權重值的分配示意圖

3. 系統設計與實作

3.1 遊戲架構與平台設計

Stendhal[1]是由 Marauroa[6]遊戲引擎所開發的多人角色扮演線上遊戲(Massive Multiplayer Online Role-Playing Game，MMORPG)，是一種利用地圖來將玩家群聚的遊戲類型，屬於單一遊戲伺服器架構。其設計如圖 4 所示，Stendhal 遊戲地圖並不是由一張大地圖切割而成，而是疊層或跳耀式的方式呈現，玩家透過傳送點的方式轉換到不同地圖去進行遊戲，在地圖中看似相鄰的地圖，卻是邏輯上的相鄰並非真正的相鄰，所以當玩家位處地圖 A 時，玩家只會得知位地圖 A 上的資訊。

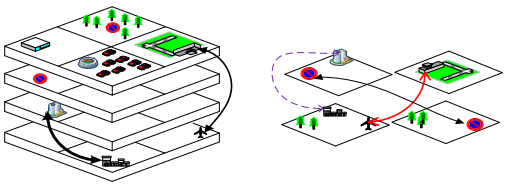

圖 4 Stendhal 遊戲設計概念圖

為了使遊戲負載盡可能有效地分攤到不同的 Back-end Server 上，我們將遊戲世界做邏輯分割，交由不同的 Game Server 負責。當玩家要進行轉換地圖時，希望讓玩家在不知情的情況下，透過在論文[2]中所設計的 Game Connection Handoff 機制，將玩家的遊戲狀態轉移到另一台 Back-end Server 上，達到玩家通透(Transparent)的特性。

3.2 遊戲負載分配策略

在論文[2]中，作者於 LVS-CAD 平台上開發了 GLARD 負載分配策略，在論文[4]中我們改善 GLARD 在動態的地圖分配時的限制，提出 GSLARD 的動態分配策略，讓相鄰的地圖盡可能交由同一台伺服器來服務，減少遊戲連線的移轉。

由於動態的分配策略，是當玩家進入遊戲時才開始分配地圖到伺服器上，仍有可能造成伺服器所分配的地圖較分散或不平均，而靜態分配會有玩家都集中在某一張地圖造成熱點地圖產生的問題。因此，我們進一步提出結合動態分配與靜態分配優點的混合分配策略，我們首先根據整個遊戲地圖的相鄰性，事先對地圖做了分配，當玩家登入後，再依玩家分佈的情形使用 GSLARD 動態的調整地圖的分配。其目的是儘量保持伺服器所負責的地圖能維持其相鄰性和地域的完整性，以減少遊戲連線的移轉和提供更好的負載平衡，使 LVS-CAD 內伺服器的資源能夠被充分利用，使負載能夠達到均衡。

3.2.1 考慮地圖的相鄰性及權重值的混合分配策略(HyGSLARD/AW)

此混合分配策略是在遊戲伺服器叢集啟動時，根據地圖權重值(即地圖的相鄰地圖張數)，先計算平均每台 Back-end Server 所能服務的地圖權重上限值，使用 Breadth-First Search(BFS)演算法事先將地圖分配到各 Back-end Server 上。當玩家登入時，則可以直接透過 Front-end Server 查詢 GSLARD TABLE，得知玩家所要進入的位置是由哪台 Back-end Server 所負責，把玩家的遊戲訊息導向該 Back-end Server 上，若當地圖上沒有玩家時，此張地圖仍會自該 Back-end Server 中釋放，等待下次玩家要進入此張地圖時，使用 LWLC 分配來將地圖分配到與此張地圖具有地域性的 Back-end Server。

考慮地圖權重值的混合分配策略的概念圖如圖 5 所示，在圖中，我們設定可服務的地圖權重值為 5，在第一次分配地圖時選定地圖 8 來做第一次分配到 Server，Server 所服務的權重值為 3 小於可服務的地圖權重值，所以可以再接受地圖的配置。然而，我們會優先考慮以地圖 8 的相鄰地圖 4、7、12 做分配地圖的選擇，在第二次分配分配地圖時，我們是選到地圖 12 來做分配到 Server 的動作，此時 Server 所服務的地圖權重值為 6，已超過可服務的地圖權重值，因此 Server 無法再接收其它地圖的配置，直到所有 Server 都將地圖分配完畢，則考慮地圖權重值的混合分配策略才結束執行，等待玩家進入遊戲後，便開始執行 GSLARD 機制。

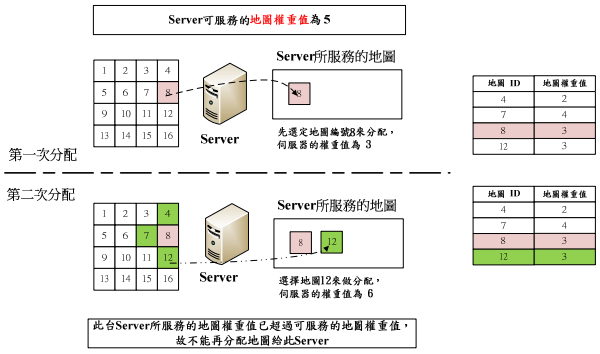

圖 5 考慮地圖的相鄰性及權重值的混合分配策略

3.2.2 考慮地域性的混合分配策略(HyGSLARD/A)

此節所提出的事先的地圖分配機制只考慮地圖的地域性，以相鄰地圖優先做分配，實作此分配

方法時，我們根據 Map-Adjacency Matrix，該 Matrix 中記錄著各張地圖的相鄰地圖資訊，以及每台伺服器都有可服務地圖張數的設定，互相搭配將地圖分配到伺服器上，由於此方式並沒有考慮到地圖權重值，所以有可能分配到伺服器上的地圖的權重都是最重或最輕的情形產生。

分配方式如圖 6 所示，在圖中我們得知 Server 可服務的 2 張地圖，在第一次分配地圖時，我們選定地圖 8 來做第一次分配到伺服器的動作，而我們是只考慮地圖的相鄰地圖優先分配，故以地圖 8 的相鄰地圖 4、7、12 做分配地圖優先的選擇，在第二次分配分配地圖時，我們是選到地圖 12 來做分配的動作，此時 Server 服務已滿無法再接受配置。

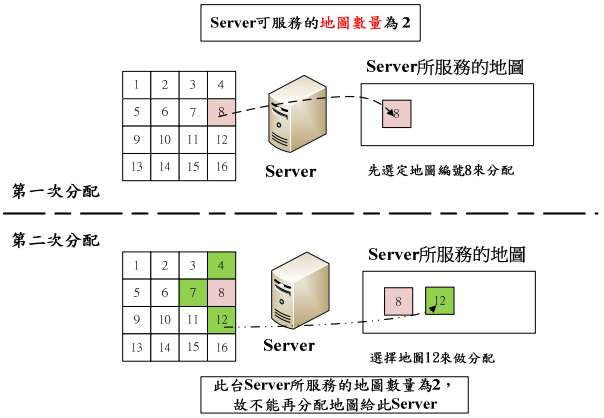

圖 6 考慮地域性的混合分配策略的概念圖

3.2.3 平均分配的靜態策略

此節所提的平均分配靜態的策略是完全沒有考慮到地圖的地域性及地圖的權重性，只是把地圖平均分散到各個 Back-end Server 上。由於我們所實驗的策略都是動態分配與混合分配的策略，因此我們亦實作此靜態分配策略來與我們所提的分配策略做效能的比較。而此平均分配靜態的方式是地圖一旦被分配到 Back-end Server 時，則不會被釋放或重新分配，並且事先將地圖分配到 Back-end Server。

我們主要是先取得伺服器的數量與地圖的張數，並且對於每台伺服器都給予一個編號，例如 Server 1 的編號即為 1，根據地圖編號與伺服器數量做餘數的計算，所計算出來的餘數和伺服器的編號相符，則將該地圖分配到此編號的伺服器上，所以可以發現到每一台伺服器所處理到的地圖皆不相鄰，伺服器所負責的地圖張數都很均勻。

4 實驗環境與效能評估

我們在 LVS-CAD 平台上實現我們所提出的負載分配策略，主要是修改 IPVS-CAD Module 於 Linux Kernel 2.6.18 中，包含對 Front-end Server 及 Back-end Server 的實作。我們實驗在大量玩家連線的情形下，比較各個負載平衡策略的效能。

4.1 實驗環境與實驗地圖

我們實驗所使用的 LVS-CAD 叢集系統平台如圖 7 及表 1，包含 1 台負責分配玩家連線的 Front-end Server，1 台提供 Game Server 使用的 MySQL Database Server，七台同質性的 Back-end Server，做為 Stendhal 遊戲的 Game Server，及八台 Client。

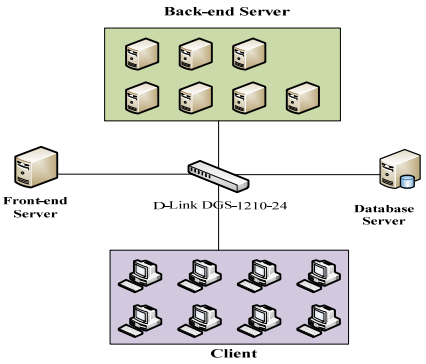

圖 7 實驗環境平台

表 1:軟硬體環境

	Front-end Server	Back-end Server	Database Server	Client
Processor	Intel Pentium 4 3.4G	Intel Pentium E2180 2.0G	Phenom II X2 545 3.5G	Intel Pentium E6700 3.2G
Memory	DDR 1GB	DDR2 2GB	DDR 4GB	DDR2 2GB
OS/ Kernel	CentOS 5.5 Kernel 2.6.18.1	CentOS 5.5 Kernel 2.6.18.1	CentOS 5.5 Kernel 2.6.18.1	Windows XP SP3
IPVS-CAD	1.2.1	X	X	X
Java Machine	X	JRE 1.6.0.21	X	JRE 1.6.0.21
Game software	X	Stendhal 0.87.1	X	Stendhal 0.87.1
Web / Database	X	X	Apache 2.2.3 PHP 5.1.6 MySQL 5.077	X
Monitor	X	Ganglia-gmond	Ganglia-gmond	X

在實驗時，必須要監控 Game Server 的負載與 Throughput，因此我們在 Back-end Server 與 Database Server 上安裝 Ganglia Monitoring System [7]監測軟體，用來蒐集 Game Server 的即時資訊。我們實驗時使用論文[2]中所設計的 Agent-Client 程式與實驗地圖。Agent-Client 主要是模擬玩家進行遊戲的行為，會發送大量的遊戲封包給 Game Server。而 Game Server 會因為玩家的行為，傳送玩家所需要的資訊，因此 Game Server 的遊戲負載也隨之增加。

4.2 實驗設計與效能評估

我們實驗了七種負載平衡策略，包含了我們的 GSLARD 與 GLARD 兩種動態的負載平衡機制，以及兩種混合策略，其一是使用依地圖的相鄰性及權

重值做事先分配，遊戲進行後使用 GSLARD 動態分配的混合策略(HyGSLARD/AW)，另一則是使用地圖的地域性做事先分配，遊戲進行後再結合 GSLARD 的動態分配的混合策略(HyGSLARD/A)。

我們另外實驗了三種靜態策略，其一是平均分配方法(MOD)，靜態平均分配方式是根據伺服器的數量與遊戲地圖的總數，計算出伺服器可以服務幾張地圖的方法，地圖一旦分配後就無法再重新分配。其於兩種分別是根據依地圖的相鄰性做事先分配的方法(Static A)與依地圖的相鄰性與權重性做事先分配的方法(Static AW)，其目的是當遊戲進行後，是否有使用 GSLARD 動態分配而使系統效能有所差異，因此分別與 HyGSLARD/A 及 HyGSLARD/AW 混合策略做效能比較。

4.2.1 各分配策略的 CPU 負載情形

在我們的實驗平台上單台 Back-end Server 所能服務的玩家人數約 500 人，因此在實驗使用 Agent-client 創造 3000 個玩家，將玩家登入到 7 台 Back-end Server 上，我們實驗的是採用一次一百人登入遊戲的方式，直到 3000 個玩家全部登入為止，且觀察當 3000 個玩家都進入 Game Server 中，各分配機制的整體伺服器平均 CPU 使用率情形如圖 8。

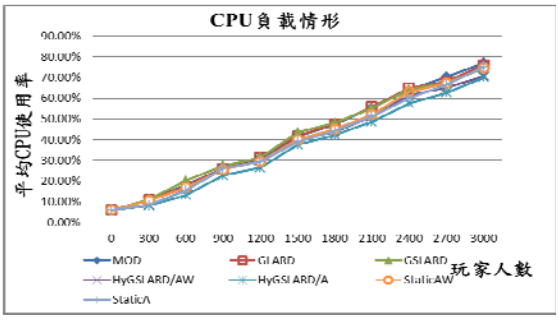

圖 8 伺服器叢集的平均 CPU 使用率

實驗中顯示，GSLARD 負載機制比 GLARD 負載機制在整體伺服器的平均 CPU 使用率情形還要好 3%至 8%。而在具事先分配的 GSLARD 混合策略上，HyGSLARD/AW 與 HyGSLARD/A 在整體伺服器的平均 CPU 使用率情形上，比 GLARD 負載機制還要好 4%至 10%，比 GSLARD 負載機制還要好 2%至 6%。而靜態平均分配機制，則因此機制不會重新分配地圖的緣故，故 CPU 負載相對的比其他分配機制還要來的高一些。

4.2.2 玩家 Handoff 次數情形

玩家的行為是難以預測的，如果玩家要轉換至不同伺服器所服務的地圖時，系統需執行 Game Connection Handoff 來轉移遊戲連線，當次數過於頻繁時，則會對伺服器叢集整體的效能會造成一定的影響。在我們的實驗中，我們記錄了系統執行 Game Connection Handoff 機制的次數，來比較不同的分配策略對整體的伺服器叢集系統效能的影響，次數越少表示越有效率，實驗結果如圖 9 所示。

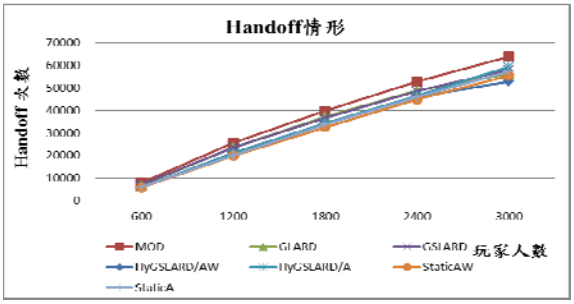

圖 9 Game Connection Handoff 次數的比較

實驗結果顯示屬於事先分配中的靜態平均分配方式(MOD)的 Game Connection Handoff 的次數相對的比其他分配機制還要高，是因為該靜態平均分配機制的伺服器，其所服務的地圖之間幾乎都沒有關係，即沒有相鄰，因此會造成玩家前進到入口點要轉換地圖時，系統都必須要執行 Game Connection Handoff 機制才能夠將玩家的遊戲連線轉換至另一個伺服器所服務的地圖上。因此在實驗中，靜態平均分配機制的 Handoff 次數都是最高的。

4.2.3 伺服器叢集 Throughput 情形

由於 Stendhal 遊戲的地圖是感知的範圍，只有在相同地圖上的玩家才能得知該張地圖上所有的資訊，因此，當玩家在某張地圖上，伺服器必須傳送該張地圖上的所有資訊給玩家。然而當有大量玩家在相同地圖時，會導致 Throughput 增加，而玩家在執行 Game Connection Handoff 機制時與伺服器載入或釋放地圖時，都會影響伺服器叢集的 Throughput，圖 10 表示伺服器叢集的 Throughput。

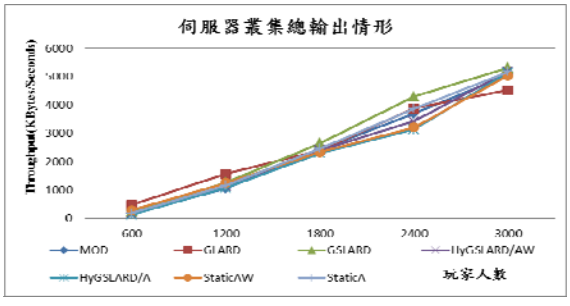

圖 10 伺服器叢集的 Throughput 比較

圖中顯示 GLARD 負載機制在玩家人數約 2400 人時，因負載不平均讓玩家感到延遲或斷線，導致總輸出量的逐漸下滑。而其他分配機制的輸出量卻能夠繼續增加，表示伺服器叢集中的 Server 資源都能被充分利用。因此在論文中所設計的分配機制，明顯改善 GLARD 負載機制因地圖分配不均所造成

整體系統負載不平均的情形，也減少玩家斷線與感到延遲的跡象，讓整體系統能服務更多的玩家。

4.2.4 玩家的分佈情形

玩家的分佈情形與伺服器的負載情形是緊緊相關的，當大量玩家同時間擠到某張地圖上，而讓該地圖成為熱點(Hotspot)地圖，會使得負責該張地圖的伺服器的負載大幅增加，負載不平衡下導致各伺服器的資源無法充分地被利用。在實驗中，我們發現到各個分配機制的玩家分佈情形，會因為該伺服器所服務的地圖張數以及玩的家行為，導致每次實驗各伺服器所服務的玩家人數差異甚大。

我們記錄了其中兩次的實驗結果，當 3000 個玩家全數都登入 Game Server 中，我們記錄每台 Game Server 的玩家分佈情形，圖 11 計算玩家於各伺服器的分佈的標準差，用來判斷各分配機制的分配效率是否能讓伺服器的負載均衡。

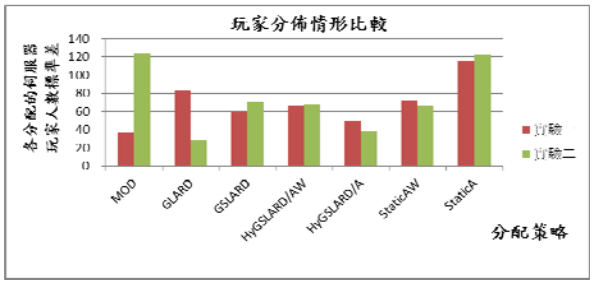

圖 11 玩家在各分配策略上的分佈差異

我們可以明顯發現靜態平均分配機制(MOD)的玩家分佈情形，在兩次實驗中的差異性非常的大，這是因為玩家的行為所導致的差異性。而 GLARD 玩家分佈的差異性，雖不像靜態平均分配機制來的懸殊，但也是有一定的差距，是因為兩次實驗中地圖分配至伺服器的情形不同。而 GSLARD 動態機制與 HyGSLARD/AW 混合這兩種機制，卻是能夠讓玩家較平均的分佈到各個 Game Server，它們參考地圖的資訊以及伺服器所能服務的總地圖權重值，盡可能讓相鄰的地圖讓同一台伺服器負責，使各伺服器所服務的地圖張數都差不多，這樣伺服器的資源就能夠充分地被利用。

從實驗中得知，GSLARD 負載機制不管在 CPU 負載或者 Throughput 等實驗結果，都明顯得比 GLARD 還要來得好，是因為 GLARD 有地圖分配過於分散情形與無法重新分配地圖的這兩種問題限制，經常會造成負載不平均的發生，而 GSLARD 改善了 GLARD 的地圖分配過於分散情形，也改善無法做到重新分配地圖的限制。而我們所提的 HyGSLARD/AW 與 HyGSLARD/A 這兩種混合策略，不僅可以事先將地圖分配到伺服器上，也能夠重新調整地圖的分配，讓每台伺服器所服務的地圖盡可能達到具有相鄰性，且地圖權重值又不會超過伺服器可服務的總地圖權重值，讓整體伺服器的資源能夠被充分地利用。靜態平均分配方式(MOD)則是在沒有考慮地圖權重值與地圖之間的相鄰性時，對於整個伺服器叢集都會造成很大的負擔，尤其會因為玩家轉換地圖而系統執行 Game Connection Handoff 機制的次數過於頻繁，造成 Front-end Server 與 Database Server 的 Overhead 過重，讓整體系統效能變差。而 StaticA 與 StaticAW 這兩種靜態分配，因地圖無法動態釋放的關係及玩家行為，導致玩家會集中在某些伺服器上，尤其是 Static A 更明顯，而與有使用動態 GSLARD 混合策略比較之下，StaticA 與 StaticAW 使系統效能較差。

5 結論

我們在 Linux Kernel 上設計且實作結合動態 GSLARD 地圖分配與靜態地圖分配優點的混合策略於 LVS-CAD 叢集式系統平台上。其中 GSLARD 的特點是盡可能讓具相鄰性的地圖交由同一台 Game Server 負責，減少執行 Game Connection Handoff 來轉換玩家的遊戲連線與存取遊戲資料庫的次數，降低整體系統負載。而在我們所設計與實作的混合策略中，我們設計兩種事先地圖分配方法，在遊戲開始前，依照地圖之間的相鄰性及權重，將地圖事先分配到各伺服器。而在遊戲開始後，當地圖上沒有玩家時地圖會被動態地釋放，之後再使用 GSLARD 策略來分配。

實驗結果顯示，同屬動態地圖分配的負載分配機制，GSLARD 的平均 CPU 使用率比 GLARD 還要低 8%。而動態的 GSALRD 加上事先地圖分配的混合類型的負載機制，比單純動態的 GSLARD 與 GLARD 還要各低約 6%、10%。因此，事先將地圖依地圖的資訊分配到適當的 Game Server 上，不僅能降低整體遊戲伺服器叢集的平均 CPU 使用率，也能夠使地圖盡可能地保持完整性，減少 Game Server 都處理不相鄰的地圖，降低地圖破碎性的發生。

參考文獻

[1] “A Multiplayer Online Role Playing Framework to develop games - Stendhal”, http://arianne.sourceforge.net/game/stendhal.html.

[2] 俞博文，姜美玲，李政翰，“基於地圖分割實作可延展性的負載平衡伺服器叢集於大型多人線上遊戲”，全國計算機會議, Taiwan, R.O.C., Dec. 2-3, 2011.

[3] H. H. Liu, M. L. Chiang, and M. C. Wu, “Efficient support for content-aware request distribution and persistent connection in Web clusters”, Software Practice & Experience, vol. 37, pp. 1215-1241, 2007.

[4] 李政翰，“基於地域性的需求分配策略以提供負載平衡於支援大型多人線上遊戲的伺服器叢集”，國立暨南國際大學資訊管理研究所，碩士論文, 1 月, 2013.

[5] “The Linux Virtual Server Project - Linux Server Cluster for Load Balancing”, http://www.linuxvirtualserver.org/.

[6] “A Multiplayer Online Role Playing Framework to develop games - The Arianne Project”, http://arianne.sourceforge.net/engine/marauroa.html, 2011.

[7] “Ganglia Monitor System”, http://ganglia.sourceforge.net/.