2025年臺灣國際科學展覽會
優勝作品專輯

作品編號　190004

參展科別　電腦科學與資訊工程

作品名稱　沙盒類遊戲式學習平台系統伺服器架設節能效率研究：以Minecraft為例

就讀學校　臺北市立大同高級中學
指導教師　王昱珺
　　鄭福炯
作者姓名　許紀帆

關鍵詞　Minecraft伺服器、記錄型擴充套件、使用者分析

作者簡介

我是就讀市立大同高中高二的許紀帆，我從高一開始才發現我對資訊有興趣，也是從這個時候才開始寫程式，吽爾會參加一些小比賽跟檢定。這次是我人生第一次接觸科展，在老師和教授的協助下我完成了第一個專案，也讓我學到除了競程以外的其他資訊領域。特別感謝王昱珺老師與鄭福炯教授的用心指導，讓我有機會參加這次的2025國際科展。未來我將會學習競賽程式與專案實作的各種知識與技巧，並繼續投入資訊領域的研究。

2025年臺灣國際科學展覽會
研究報告

區別：北區

科別：電腦科學與資訊工程科

作品名稱：沙盒類遊戲式學習平台系統伺服器架設節能效率研究：以Minecraft為例

關鍵詞：Minecraft伺服器、記錄型擴充套件、使用者分析

編號：
（編號由國立臺灣科學教育館統一填列）

中文摘要

本研究以 Minecraft 為例，探討沙盒類遊戲式學習平台系統伺服器架設的節能效率，旨在透過動態調整伺服器數量降低總CPU使用率，提升伺服器的管理效能和能源使用效率。隨著線上遊戲的普及，伺服器的營運管理變得越來越複雜，如何在滿足玩家需求並同時降低能源消耗成為一個重要議題。本研究將分析伺服器資源使用狀況，特別是在玩家活動量高低波動的情境下，透過管理策略的調整，探討其對節能效率的影響。

研究透過實證數據的收集與回歸分析，建立一套可應用於 Minecraft 伺服器的節能動態調整系統，並探討動態調整的具體效率。研究結果發現隨著玩家人數增加，越接近系統負載上限，節能效果會越來越不明顯，以本次研究的伺服器來講玩家人數到達35人以後就無法再減少伺服器數量。

英文摘要

This study explores energy-saving efficiency in sandbox game server management, using Minecraft as a case study. The primary goal is to dynamically adjust the number of active servers to optimize overall CPU usage, thereby enhancing server management efficiency and energy utilization. With the increasing popularity of online gaming, server management has grown more complex, necessitating solutions that meet player demands while minimizing energy consumption. This research focuses on analyzing server resource utilization, particularly under conditions of fluctuating player activity, and assesses the impact of adaptive management strategies on energy-saving efficiency.

Through empirical data collection and regression analysis, this study establishes a dynamic adjustment system for energy savings that is applicable to Minecraft servers. The system's effectiveness in reducing energy usage is further examined under varying levels of server demand. Results show that as player numbers increase and approach the system’s load capacity, the energy-saving effect diminishes. For the servers examined in this study, once the player count reaches 35, it is no longer feasible to reduce the number of active servers without compromising performance.

壹、研究動機

隨著數位技術的不斷進步，Minecraft 這類沙盒遊戲被廣泛應用在多種場合，例如教育應用(Bebbington,2014)、數學探討(Lischka,2015)…等，無論是娛樂還是專業領域，其多人連線特性和高度自由的世界構建能力，均為許多大型伺服器運行的基礎。然而，在 2030 年之前全球伺服器預計以每年10%的年增長率持續增加(IPCC, 2021)，加上伺服器長時間保持高負載運行

，尤其在遊戲內多人連線與即時運算的情況下，能源需求更是顯著增加。如何在不影響使用者遊戲體驗的前提下，達成有效的節能，已成為研究與技術實踐中的一大挑戰。

因此，本研究聚焦於Minecraft 伺服器的能源使用效率，期望在探討伺服器資源管理與負載平衡機制的基礎上，提出一套具體的節能優化方案。本研究分析在不同使用情境下伺服器的資源配置，並透過動態調整伺服器運行數量，關閉閒置的伺服器以減少不必要的能源消耗，特別是在伺服器低負載或空閒時段，最大化節省運行成本。此外，本研究也將探討如何根據玩家數量動態調整伺服器數量，實現伺服器的即時擴展或縮減，進一步優化整體能源使用。

貳、研究目的及研究問題

一、研究目的與問題

根據上述研究動機，本次研究目的與問題如下：

(一) 探討伺服器的資源使用狀況，以提升節能的效率與管理上的應用。

(二) 探討在固定伺服器硬體配置的前提下，伺服器在離峰時期關閉部分分流的節能效率為多少。

(三) 了解玩家遊玩時影響伺服器資源使用狀況之因素。

(四) 伺服器資源分配。

二、預期價值

隨著人工智慧應用的快速發展，伺服器需求持續增加，不只如此，線上遊戲市場亦日益壯大，特別是Minecraft伺服器市場。因此，本研究旨在創造以下幾項學術與實務價值：

(一) 提供管理者一個基於活躍玩家變化的伺服器運營狀況分析工具，提升伺服器管理效能。

(二) 透過開發模型計算離峰時段減少伺服器分流數量的節能效率，以達到降低運營成本的目的。

(三) 進一步優化節能策略，促進資源使用的可持續性發展。

參、文獻探討

一、Minecraft

Minecraft 中文翻譯成我的世界，以下以Minecraft 進行論述。Minecraft 是一款以高開放性 3D 沙盒類為名的電子遊戲，每月活躍人數超過一億人，並在2014年被微軟(Microsoft)收購，由一名瑞典遊戲設計師，本名 Markus Alesj Presson，人稱Notch 開發，現由微軟旗下 XBox 遊戲工作室中的 Mojang Studios維護更新，目前遊戲分為三個版本，Java 版由 Java 語言開發，基岩版和教育版由 C++開發，遊戲有生存、創造、冒險、旁觀者四種遊玩模式，整個世界都是由 3D 立體方塊組成（Minecraft 官方網站，n.d.）。

圖 1

Minecraft

註：出自 Minecraft 官網

二、沙盒類遊戲

沙盒類遊戲，英文為 sand box，原本是指一個充滿沙子的玩耍區域，在裡面可以自由的在裡面搭建任何東西，之後沙盒遊戲就被引申為自由度非常高的遊戲。沙河類遊戲最大的特點就是有極高的自由度，遊戲內沒有固定玩法，大部分沙盒類遊戲都具有開放性地圖（無邊界地圖），且沒有指定任務給玩家做，例如：Raft 與俠盜獵車手。

俠盜獵車手是一款第三人稱動作遊戲，其中最大的特點就是角色扮演以及第三人稱射擊，整個遊戲是以對美國落聖都文化為背景（Rockstar Games, n.d.）。Raft 則是一款以全球暖化海平面上升為背景的生存遊戲，玩家需要不斷的收集資源及食物、打造屬於自己的木筏，最後會找到叫做烏托邦的地方（Redbeet Interactive, n.d）。兩個遊戲雖然都有主線任務，但是整個遊戲環境為開放世界，給玩家非常大的自由度。

三、Minecraft 伺服器的種類

伺服器，英文為server，是一種專門設計的計算機系統或裝置，用於管理網路資源並向其他主機或客戶端提供服務。它們通常具有強大的處理能力、大量的存儲空間和快速的網路連接能力。伺服器有多樣類型，包括檔案伺服器、郵件伺服器、應用伺服器和資料庫伺服器等。每種伺服器根據其提供的特定功能和服務而有所不同。伺服器的特點包括高性能的硬體、高速的網路連接和安全性措施，旨在確保數據的可靠性和安全性（Monteclaro, 2023）。

伺服器是 Minecraft 中唯一可以和朋友一起遊玩的管道，透過網路傳送資料封包，再交由伺服器運算並回傳對應的動作指令。以下是根據 Mitchell Smith針對 Java 版伺服器的三大種類（Minecraft, n.d ; BisectHosting, n.d ; About Spigot, 2021 ; Minecraft Server Types, n.d）：

（一）官方提供的原版伺服器：不能裝額外的擴充套件，但非常節省資源。

（二）模組伺服器：可以安裝模組，但客戶端與伺服器端皆須安裝。在 Minecraft 中可以運行用戶創建的附加組件稱為模組。這些模組擴展了 Minecraft 原版的遊戲玩法。Minecraft 原生不支援模組，需要特殊版本才能運行。Forge 和Fabric 是兩個最受歡迎的模組化Minecraft 客戶端（Minecraft mods, n.d）。

（三）擴充套件伺服器：可以裝擴充套件在伺服器上，可以提供突破原版只限單核運作的限制。

四、資料庫的種類

資料庫是儲存和組織數據的系統，允許進行高效檢索和管理。它支援複雜查詢、事務處理、自動備份，並能擴展以適應不同規模的需求。根據使用資料的方法大致上可分為以下幾種資料庫（Oracle, n.d.）：

(一) 關聯式資料庫：使用表格、行和列來組織數據，強調數據之間的關聯和完整性，例如：mySQL。

(二) 非關聯式資料庫：適合靈活儲存非結構化或半結構化數據，如文件、鍵值對，例如：MongoDB。

(三) 分佈式資料庫：跨多個服務器或位置儲存數據，以提高數據可靠性和存取速度，例如：Cassandra。

(四) 物件導向資料庫：將數據作為對象儲存，支持對象的屬性和方法，例如：ObjectDB。

五、 機器學習應用於效能分析

(一) 機器學習在伺服器性能分析中的重要性

根據Chen研究(2021)，由於數據量的快速增長，伺服器性能優化的應用中使用機器學習變得越來越重要。研究指出，傳統的伺服器性能分析方法，如靜態規則或預定義的閾值，通常較難應對伺服器在面對突發事件或異常負載時的複雜情況。而機器學習能夠學習、預測，這樣資源管理更精確和高效(Chen, 2021)。Weng(2019)的研究也指出，機器學習在分析和預測伺服器資源使用情況時，能夠明顯的提升管理效率，特別是在應對大規模動態數據環境(Weng, 2019)。

(二) 回歸分析在伺服器負載預測中的應用

回歸分析是一種常見的機器學習技術，適合用於分析伺服器應能與預測負載。根據Lee和Park的研究，回歸模型可以將伺服器資源使用量（如CPU、記憶體等）與玩家行為數據結合，進而有效預測伺服器的資源需求。這種方法特別適用於多元回歸模型中，因為它能同時考慮多個影響因子，提高預測的準確度(Lee & Park, 2020)。然而，伺服器數據中常會出現共線性問題，這會影響模型的準確度。因此，研究建議可以使用變異數膨脹因子等方法檢測、處理共線性問題並改善模型的穩定性(Wang, 2019)。

(三) 正規化技術在伺服器性能預測中的應用

針對伺服器負載分析中的高維度資料，正規化技術被廣泛應用。Smith(2020)指出，Lasso回歸通過L1正規化技術來自動進行變數選擇，特別適合用在伺服器負載預測，因為它能夠縮減不重要的變數權重，簡化模型和提升模型的準確性(Smith, 2020)。此外，Ridge回歸利用L2正規化來解決變數之間的共線性問題，特別適合用在處理伺服器資源消耗數據。Gupta（2021）的研究進一步指出，Elastic Net結合了Lasso和Ridge的優點，能夠在伺服器負載分析中同時進行變數選擇和解決共線性問題，可以提供更穩定和準確的預測結果(Gupta, 2021)。

(四) 深度學習技術在伺服器負載分析中的潛力

隨著伺服器運行數據量的增加，深度學習技術在伺服器性能分析中逐漸顯現出其潛力。根據Hochreiter和Schmidhuber(1997)的研究，LSTM模型特別適合處理伺服器資源的動態變化，能夠捕捉時間序列數據中的長期依賴關係，從而提供精確的長期負載預測(Hochreiter & Schmidhuber, 1997)。同時，根據LeCun(1998)的研究，卷積神經網路 在識別伺服器數據的模式和異常數據有很高的效率，特別是在檢測突然的負載異常(LeCun, 1998)。

綜合上述研究，機器學習技術在伺服器性能分析中的應用涵蓋了傳統回歸分析到深度學習模型的多種方法。這些技術不僅能夠動態預測資源使用情況，還能透過處理異常負載來提高伺服器的運行效率與穩定性。

六、伺服器性能關鍵因素分析

伺服器性能測試的不同方法對於理解伺服器在各種負載條件下的表現提供了豐富的依據。Pedram 和 Hwang(2010)使用了合成工作負載生成器，模擬伺服器在各種場景下運行和負載的表現 (Pedram & Hwang, 2010）。Bohra 和 Chaudhary(2010)則利用了包括 Iozone、Bonnie++ 和 BYTEMark ...等多種測試工具，這些工具專注於測試伺服器的數據讀寫能力以及虛擬機器運行的性能(Bohra & Chaudhary, 2010）。Alan(2014)研究了數據傳輸工具如 SCP、rsync、FTP 等的應用，這有助於了解伺服器在大量數據傳輸時的性能(Alan , 2014)。Kansal(2006)進行的研究則是基於 SPECcpu 2006 和 IOmeter，主要評估了伺服器的 CPU 和 I/O 表現(Kansal, 2010)。

七、伺服器功耗

Pedram和Hwang(2014)提出在虛擬伺服器功耗與性能建模的概念，研究指出隨著工作負載的變化，伺服器負載會大幅波動，因此能夠即時調整資源對於節能很重要(Pedram & Hwang, 2014)。

在Bohra和Chaudhary(2010)的研究中特別探討針對虛擬伺服器和雲端伺服器評估功耗。他們使用了如NAS-NPB、Iozone等工具分析伺服器的各項資源消耗，結果顯示，數據的頻繁讀寫操作以及虛擬機的運行會顯著影響伺服器的總體功耗(Bohra & Chaudhary, 2010)。

Alan(2014)在研究數據傳輸和擴充套件使用對伺服器能耗的影響時發現，在伺服器環境中，rsync、scp等傳輸工具和擴充套件的使用會影響伺服器的計算性能以及增加伺服器的功耗(Alan, 2014)。

八、總結

綜上所述，伺服器性能分析和管理已逐漸依賴機器學習技術來應對動態負載和資源需求。傳統方法雖在初步監測中有效，但難以應付伺服器運行中的異常情況，像是在負載突然飆高時，固定閾值的預警系統可能無法及時反應，導致資源分配不均或伺服器過載。再者，當伺服器的性能逐漸下降或出現記憶體洩漏或網路延遲問題的時候，這些方法往往無法有效捕捉，導致問題長期積累而不被發現 (Ramesh & Dey, 2018；Jiang, Zhao, & Chen, 2015)。機器學習中，尤其是回歸分析、正規化技術和深度學習模型，提供了更準確的預測及異常偵測能力，使得伺服器在處理大規模資料時能夠更靈活應對。此外，深度學習的應用如 LSTM 和 CNN...等，能有效捕捉時間序列中的長期依賴性，進而提升負載預測的準確性和伺服器性能的穩定性。

同時，不同類型的資料庫和伺服器架構在效能與資源管理中的角色亦不容忽視，這些技術可以針對伺服器的資源消耗、數據讀寫與負載優化，提供重要支援。資料庫的選擇，依據伺服器需求，在提高數據檢索與處理速度上發揮關鍵作用。

肆、研究設備及器材

根據伺服器託管以及伺服器管理者的設備配置中，通常會以一位玩家 1GB 記憶體為最低標準，且因為看重伺服器 CPU，所以不選擇使用 CPU內建顯卡，而是使用好一點的 GPU 以支持基本伺服器圖形運算。這不僅能明顯降低伺服器卡頓，也能減少因為伺服器效能造成對玩家遊玩上的限制。在資料庫選擇使用mySQL，mySQL不只是Logger和Plan支援的資料庫之一，mySQL 更具備完整的函數及語法（Groot，2019），因此本次研究伺服器架設的硬體與作業系統開發環境如下：

一、資料統計及運算

(一) CPU : AMD Ryzen 5 5600X

(二) GPU : RX 6600xt

(三) RAM : 32GB

二、作業系統與開發環境

(一) 作業系統：Windows 11 PRO

(二) 開發環境：Visual Studio Code Python 3.11.1

(三) 資料庫：mySQL

三、Minecraft伺服器

(一) 主要大廳

1. CPU : Intel(R) Core(TM) i7-8700 CPU @ 3.20GHz
2. GPU : Intel(R) UHD Graphics 630
3. RAM : 16GB
4. SSD : 256GB

(二) 生存分流

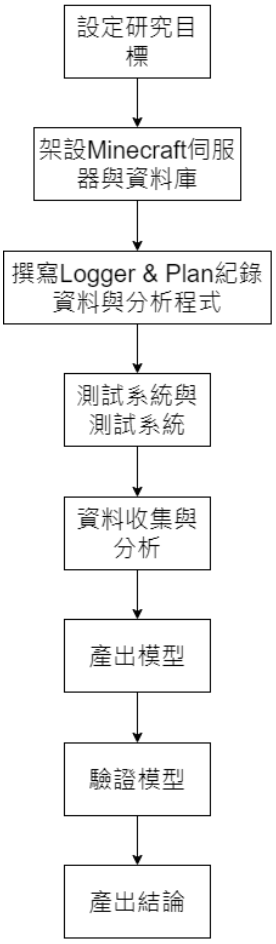
1. CPU : CPU:Intel(R) Core(TM) i7-3770 CPU @ 3.40GHz
2. GPU : IvyBridge GT2 [HD Graphics 4000]
3. RAM : 8GB
4. SDA : 500GB

伍、研究過程、方法及進行步驟

一、研究流程

圖 2

研究流程



註：由作者製作

圖2為本次研究的流程圖，本研究首先設定研究目標，並尋找相關文獻。接著進入研究階段。首先先架設Minecraft伺服器與資料庫，並持續處理擴充套件之間的相容性問題以及遊戲漏洞。接著開始收集研究所需的相關數據，研究過程中，我們使用Logger和Plan記錄數據，以確保資料的完整性與準確性。之後藉由數據進一步得出模型並驗證。最後，基於分析結果產出結論。

二、 研究方法及步驟

本研究之研究方法採用實證研究，實證研究是基於觀察和實驗來收集資料的研究方法，目的是通過科學手段驗證或反駁理論假設。它重視客觀的資料分析和結果的可重複性，以下為本次研究之步驟：

(一) 透過實際架設 Minecraft 伺服器來研究Logger 具體應用。

(二) 透過修改Logger 提取資料分析需要的資料。

(三) 利用Logger 所蒐集到的資料進行資料分析。

(四) 運用機器學習做迴歸分析產出數學模型。

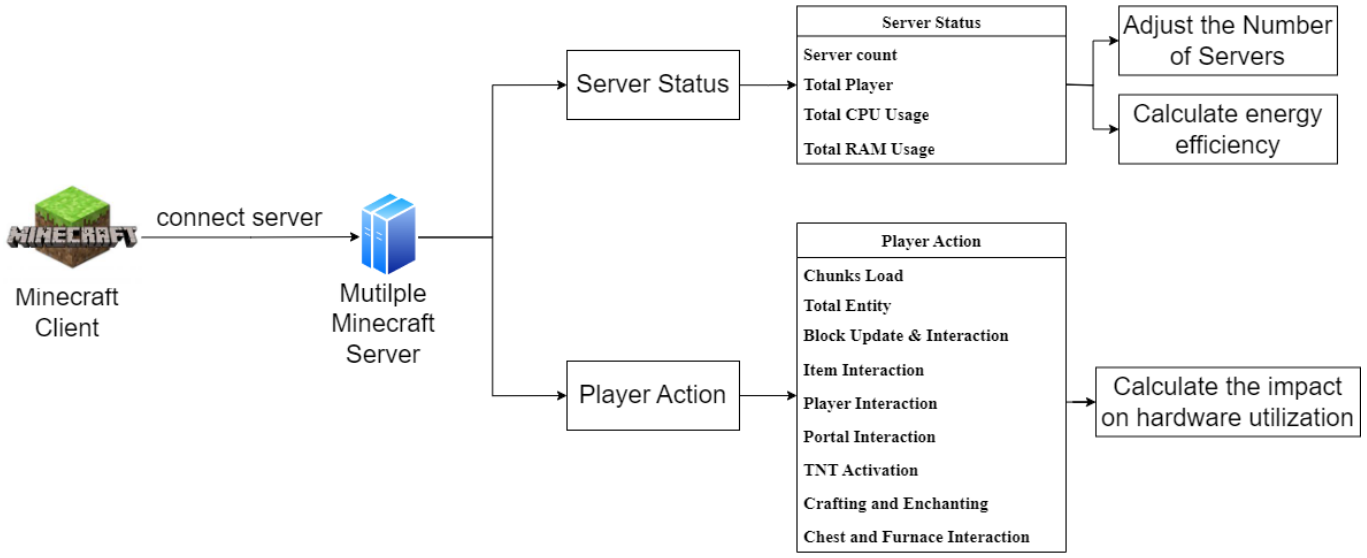
(五) 用玩家資料調整、驗證模型

三、系統架構圖

根據圖3和圖4，本研究設計了多個Minecraft伺服器。首先，玩家連接到Minecraft伺服器的Lobby，該伺服器透過BungeeCord代理系統將玩家分配至不同的生存伺服器（Survival1至Survival4）。用於分三玩家和動態調整伺服器數量。而當玩家進入伺服器遊玩的時候負責記錄數據的擴充套現分別記錄伺服器硬體相關數據和玩家行為，用動態調整伺服器數量以及找到玩家行為與硬體數據的關聯。

圖 3

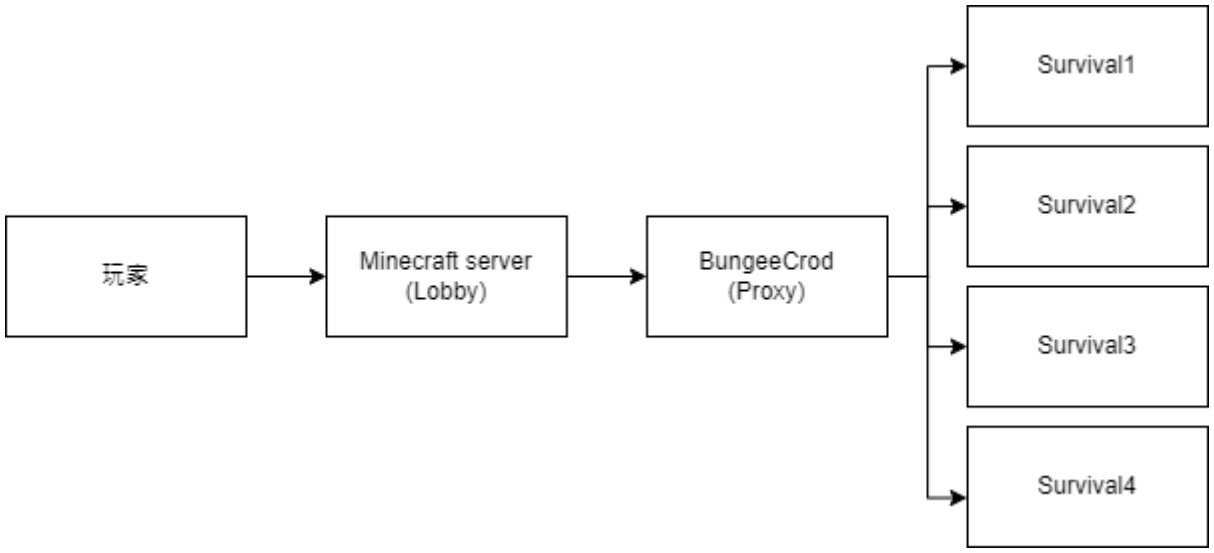
系統架構圖



註：由作者製作

圖 4

伺服器架構圖



註：由作者製作

四、Minecraft伺服器及資料庫架設

在本次研究中總共使用到5台主機分別運行1個大廳及4個生存伺服器，並用Bungeecord讓這5個伺服器得以互相連通。且為了確保在尖峰時段會有多數伺服器達到滿負載，看到尖峰與離峰時段伺服器個數的明顯差異，所以特別設定每個伺服器玩家人數上限為15位玩家，每個生存伺服器都裝有20個擴充套件，其中包含Logger以及Plan用來記錄各項資料。

五、資料統計及分析系統

(一) 資料收集及用處

本次研究使用了Minecraft伺服器的擴充套件Logger以及Plan來進行數據收集。此擴充套件能收集到伺服器的運行數據，包含不同類型的玩家行為數據及系統性能數據。

1.玩家行為數據：這類數據包括玩家在伺服器中的所有遊戲行為，主要分為玩家加入、玩家離開、死亡事件、物品掉落、物品撿拾、玩家間的對話、玩家發出的指令等。這些行為數據能夠幫助我們了解伺服器內部的玩家活動量以及玩家行為對伺服器負載的影響。

2.伺服器系統性能數據：伺服器內部的硬體監控數據對於評估伺服器運行狀況至關重要。本研究通過Plan定時收集CPU使用率、內存占用、磁碟讀寫等相關數據。這些數據能夠幫助建立伺服器資源使用情況與玩家行為之間的關聯。

3.事件類別數據：不同的伺服器事件（如TNT爆炸、建築方塊放置、傳送門創建、傳送）會引發資源使用波動。本研究將這些事件作為數據收集的一部

分，通過記錄每個事件發生的頻率，探討其對伺服器負載的影響。這類數據能夠幫助識別那些活動會引起顯著負載波動。

4.為了進一步分析這些數據，這個系統會每分鐘收集到的玩家行為和伺服器系統性能數據進行同步並儲存到資料庫中。這些數據將構建出一個時間序列，展示伺服器的負載變化及與玩家行為的關聯性。

表1

紀錄伺服器硬體數據資料表

timestamp	player_count	server_count	cpu_load	memory_usage
2024-10-26 05:01:30	6	4	72.47%	42.29%
2024-10-26 05:11:30	7	4	44.18%	57.63%
2024-10-26 05:15:30	9	4	72.85%	54.78%
2024-10-26 05:23:30	13	2	123.25%	65.28%
2024-10-26 05:34:31	16	2	150.70%	84.93%
2024-10-26 06:44:32	13	2	86.21%	90.18%
2024-10-26 06:49:32	20	2	149.35%	97.10%
2024-10-26 07:19:33	19	2	108.75%	92.85%
2024-10-26 07:39:47	0	1	9.85%	28.73%
2024-10-26 07:45:47	5	1	42.15%	52.26%
2024-10-26 07:48:47	11	2	88.13%	59.61%

註：由作者製作

(二) 統計分析及方法

在本研究中，我們不僅關注Minecraft伺服器的運行狀況，還希望透過分析玩家行為及伺服器的硬體性能數據來找出兩者之間的相關性。因此，針對不同類型的數據，本研究採用多種統計分析方法，並加上機器學習技術，以達到未來能透過預測調整伺服器數量。

1.資料處理

在數據分析之前，首先對收集到的伺服器數據進行處理。由於伺服器運行過程中可能會有異常數據或空缺數據，如玩家短暫離線或伺服器關閉，這

些數據可能對統計分析產生影響。因此，我們對所有數據進行了篩選，移除明顯不合理的數據值。

2.時間序列分析

由於伺服器數據的收集是基於時間序列的方式進行的，因此使用時間序列分析技術來理解伺服器性能與玩家行為之間的動態關係。透過對每分鐘的CPU使用率、內存占用率等數據進行分析，使能夠識別出伺服器負載的高峰時段，並將這些高峰與同時發生的玩家行為事件進行對比分析。這種方式可以幫助我們找出可能引發伺服器負載增加的具體玩家行為。

3.回歸分析

為了定量分析玩家行為對伺服器硬體負載的影響，本研究採用了多種回歸分析。通過將伺服器的硬體性能指標作為應變數，並將各類玩家行為事件作為自變數，我們構建了多個回歸模型，以評估不同行為對伺服器負載的影響程度。

4.加權平均法

在進行資料分析的過程中，針對伺服器運行負載的各類數據，採用了加權平均法來計算整體的伺服器效能狀況。由於不同的伺服器事件對伺服器的影響不同，因此我們給每類事件分配了不同的權重，並根據這些權重來計算每一分鐘內伺服器的總負載。權重的分配是基於多元回歸分析的結果，通過對過去的數據進行反覆調整，以確保能反映真實的伺服器運行情況。

5.交叉驗證法

本研究採用了K折交叉驗證（K-Fold Cross Validation）來檢驗模型的穩定性，將數據集隨機分成10個子集，每次選擇其中一個子集作為測試集，其餘子集為訓練集，進行多次交叉訓練和測試，最後獲得預測的誤差值，並使用均方誤差法評估其準確性。

六、動態調節伺服器數量擴充套件

為了達到讓伺服器能在低負載時減少伺服器數量已達節能效果，本研究開發了一個能夠動態調節伺服器數量的擴充套件，以便自動化管理伺服器負載並提升資源使用效率。系統分別透過 Spigot 和 Bungeecord 的API協同運作，達到動態調整的效果。以下為各部分的技術應用介紹。

(一) Spigot

1.監控效能：利用伺服器資源監控工具，定期記錄CPU和記憶體使用率等伺服器硬體使用。系統會根據這些指標判斷是否需要增加或減少伺服器數量。

圖 5

獲取伺服器硬體數據及總人數

```
#spigot
public double getCpuLoad() {
return osBean.getSystemCpuLoad();
}

public double getMemoryUsage() {
long freeMemory = osBean.getFreePhysicalMemorySize();
long totalMemory = osBean.getTotalPhysicalMemorySize();
return 1 - ((double) freeMemory / totalMemory);
}
```

```
# Bungeecord
int totalPlayers = plugin.getProxy().getOnlineCount();
AtomicInteger onlineServersCount = new AtomicInteger(0);
AtomicInteger processedServers = new AtomicInteger(0);
int totalServers = plugin.getProxy().getServers().size();
```

註：由作者製作

2.判斷增加或減少伺服器：當伺服器負載或玩家人數超過設定的臨界值時，系統自動啟動新的伺服器來分擔負載，並透過Bungeecord上的Player Balancer把當前玩家均勻分配至所有伺服器。

圖 6

動態調整伺服器數量

```
public boolean startServer(String serverName, String host, int port) {
String sshHost = plugin.getConfig().getString("ssh.host", "localhost");
String sshUser = plugin.getConfig().getString("ssh.user", "user");
String sshPassword = plugin.getConfig().getString("ssh.password", "password");
int sshPort = plugin.getConfig().getInt("ssh.port", 22);
String startCommand = plugin.getConfig().getString("ssh.start_command", "bash
/path/to/start_server.sh");

String command = startCommand + " " + port;

return executeRemoteCommand(sshHost, sshUser, sshPassword, sshPort, command);
}
```

註：由作者製作

3.資料庫：效能和負載數據會定期匯入資料庫，用於節能效率分析。

(二) Bungeecord

1. 負載平衡：使用Player Balancer這個擴充套件，在每次伺服器數量更動的時候平均分配當前玩家到所有伺服器，降低單一伺服器的壓力。

2. 與Spigot擴充套件溝通：當需要新增或刪除伺服器時，Bungeecord 會同步通知所有伺服器，確保所有玩家連接都能被正確分配至預期的伺服器。

圖 7

透過BungeeCord指揮調節伺服器數量

```
1. private void handleClient(Socket clientSocket) {
2. try (BufferedReader in = new BufferedReader(new
   InputStreamReader(clientSocket.getInputStream()));
3. PrintWriter out = new PrintWriter(clientSocket.getOutputStream(), true)) {
4.
5. String[] parts = in.readLine().split(";");
6. String action = parts[0];
7. String serverName = parts[1];
8.
9. if ("add".equalsIgnoreCase(action)) {
10. plugin.addServer(serverName, parts[2], Integer.parseInt(parts[3]));
11. out.println("伺服器已添加：" + serverName);
12. } else if ("remove".equalsIgnoreCase(action)) {
13. plugin.removeServer(serverName);
14. out.println("伺服器已移除：" + serverName);
15. }
16. } catch (IOException e) {
17. e.printStackTrace();
18. }
19. }
```

註：由作者製作

(三) 生成圖表與數據分析

本研究使用 Python語言將伺服器數據化成圖表，用於分析效能變化，以了解節能效率。使用時間序列圖將玩家數量、CPU 負載和記憶體使用率關係做為可視化圖表，可以用來辨識伺服器高負載及低負載的變化與對應時間，評估在不同時間所需的伺服器數量。此外，研究還透過關聯分析生成玩家數量與 CPU 和記憶體使用率之間的散佈圖，探討玩家數量與硬體之間的關係，並結合這些數據進行節能效率的比較計算。

七、受試者招募

為了研究Minecraft 記錄型擴充套件(Logger)對伺服器資源使用和玩家行

為的影響，本次研究預計將招募100位活躍的Minecraft玩家。這些玩家將在研究期間持續參與伺服器活動，確保數據的穩定性和可靠性。在招募過程中，將確保玩家明白研究目的和過程，並自願參加。此外，研究將嚴格保護參與者的隱私和數據安全，以符合倫理和法律規範（Smith & Johnson, 2021）。

玩家的招募將集中於那些能夠保持高活躍度的玩家，這樣的設置有助於獲取精確且有代表性的數據，以便深入分析Logger擴充套件對伺服器資源管理和玩家行為的影響。研究期間，玩家的參與度將被密切監控，以確保數據的有效性和研究結果的可靠性。

陸、研究結果

本次研究共收集了 54 位參與者的數據，其中年齡分布集中於18歲以下共49位，其次是 18 至 20 歲共計3位以及 21 至 30 歲共2位玩家。在性別分布上，共 52 位參與者為男性，女性則有 2 位。對於 Minecraft 的熟悉度自評(滿分為5分)，大部分參與者評為 4 分或 5 分，分別有 21 位和 20 位參與者。

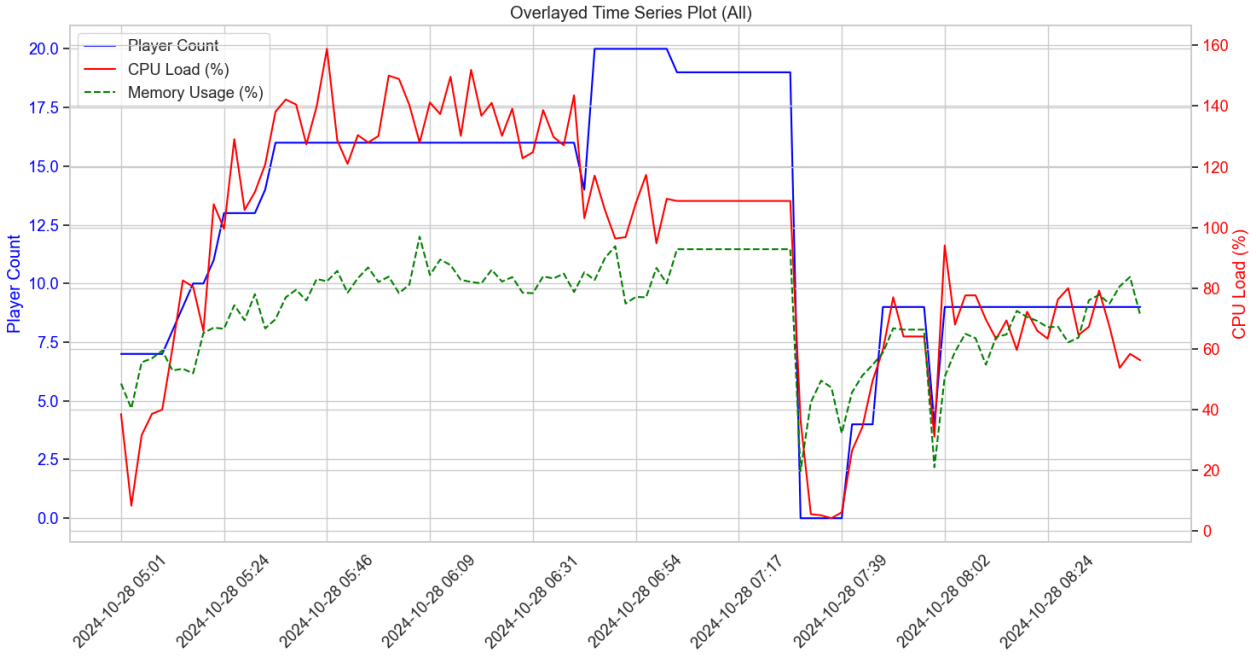
一、玩家數量與CPU使用率、RAM使用率的關係

參見圖8，在CPU負載、RAM使用率的疊加時間序列圖中，雖然沒有在伺服器崩潰或是網路傳輸受阻礙的時候把不正確的數據刪除，而是使用最近一次的數據，但還是可以很明顯的發現，隨著玩家數量增加，CPU 負載和記憶體使用率都會隨之上升，而且隨著玩家人數的變多，伺服器的負載變化也會更明顯，並且當玩家數量驟減時，CPU 與記憶體的使用率也會跟著迅速下降。

參見圖9的平面圖與圖10的3D圖，從 CPU 與記憶體使用率的相關性圖中也可以看出，在去除不正確的數據後，CPU 和記憶體的使用率有現出隨著玩家數量增加而同步上升的趨勢，與上一張圖表有相同的結果。在這張表中可以發現在目前的硬體配置下，發現玩家人數超過12人時就有可能超過一台伺服器主機可以負荷的上限，因此在玩家人數到達12人時就應該透過動態調整增加伺服器數量。

圖 8

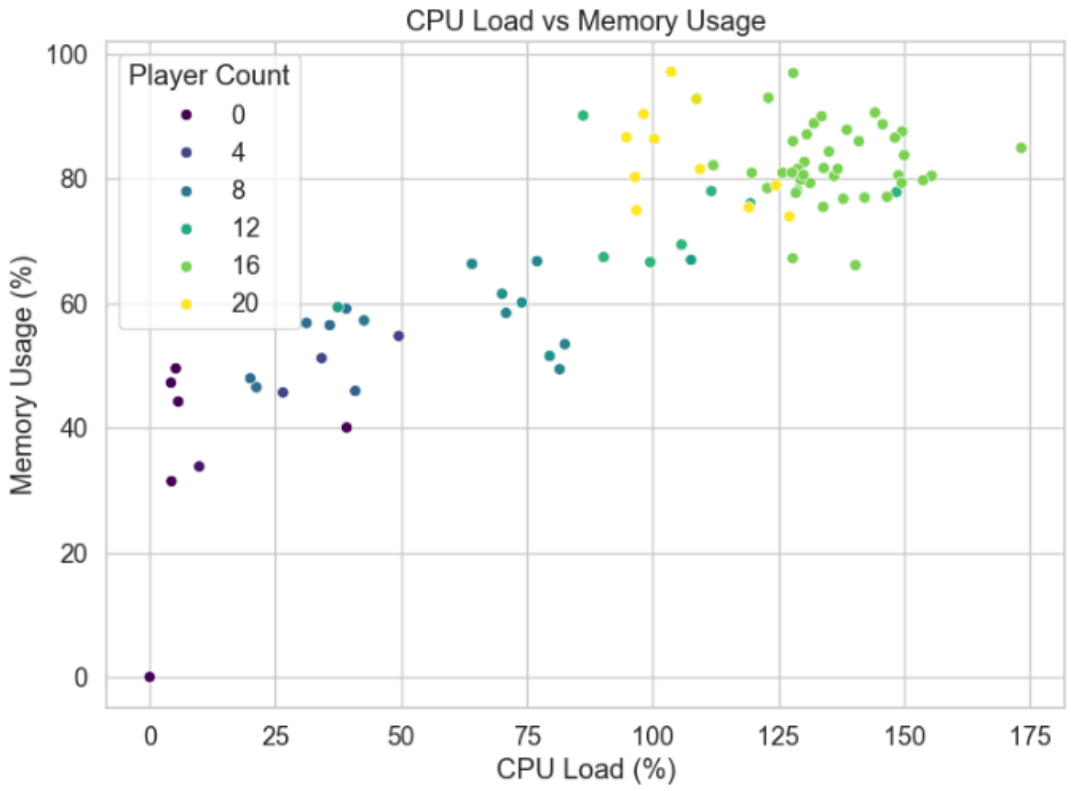
CPU負載、RAM使用率的疊加時間序列圖



註：由作者製作

圖 9

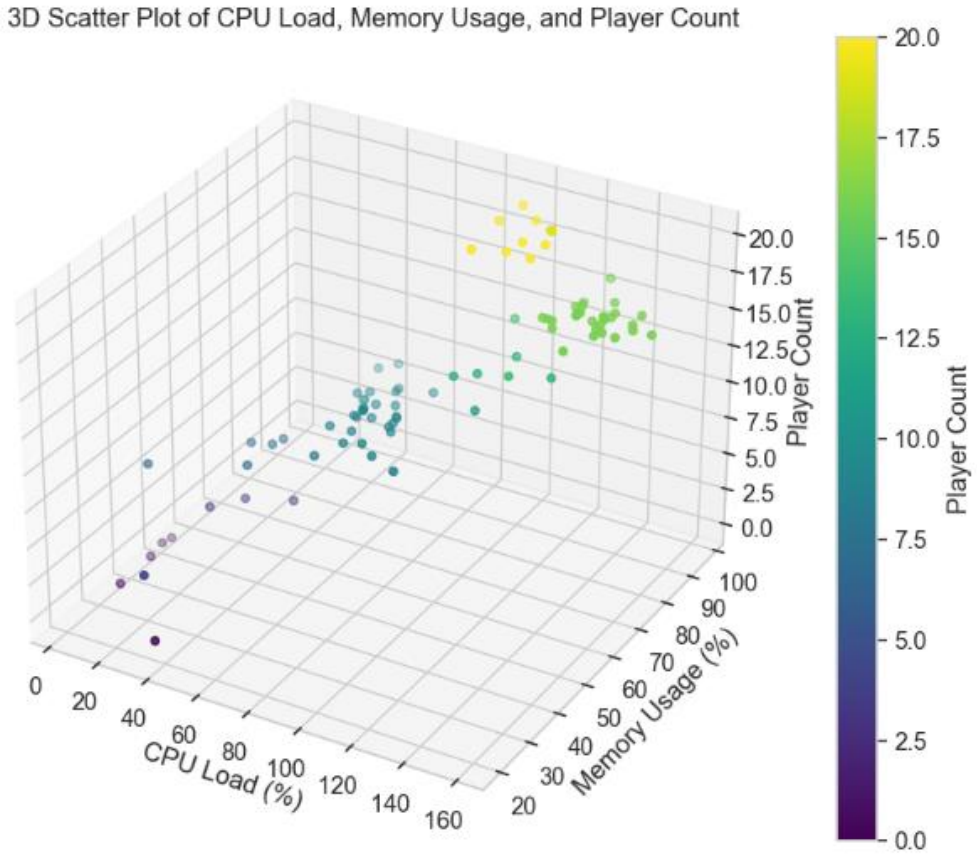
玩家人數與硬體使用率的相關性



註：由作者製作

圖10

玩家人數與硬體使用率的相關性 -3D

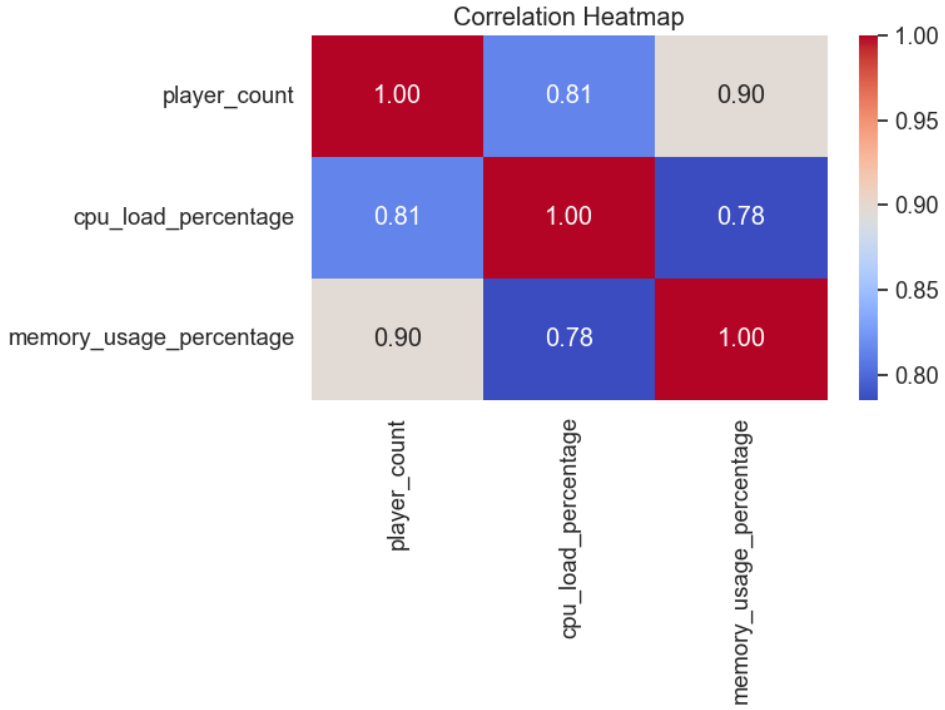


*註：*由作者製作

參見圖11，在此相關性熱力圖中，我們可以發現在這次研究中玩家數量與記憶體使用率之間的相關係數為 0.9，表示玩家數量增加會直接影響到CPU使用率。此外，玩家數量與 CPU 負載之間的相關係數為 0.81，同樣有著很高的正相關關係。在圖中還可以觀察到，CPU 負載與記憶體使用率的相關係數為 0.78，表示隨著 CPU 負載增加，記憶體使用率通常也會跟著上升。

圖 11

相關性熱力圖



註：由作者製作

二、使用動態調整伺服器數量後節省CPU使用率

參見表2、圖12與圖13，透過玩家人數與CPU節能對照表可以發現，隨著玩家數量的增加，伺服器的 CPU 節能效益會越來越不明顯。當玩家數量在 0 至 5 人時，節能效率最明顯，平均每一個伺服器最多可以減少10~12%的CPU使用率，玩家數量在 10 到 20 人之間時，節能效率已經降至 10% 以下，在玩家數量到達 30 到 35 人後，節能效益基本上趨於穩定並接近 1%，當玩家人數到35人以後因為已經無法用三台伺服器承受所有玩家，所以和原本沒有做動態調整的CPU使用率會一致。這說明在低負載時動態關閉部分伺服器可以降低能源消耗，達到優化運行成本的目的，但在高負載情況下，即便運行動態伺服器調整機制，節能效率也十分有限，伺服器已達到最佳運行容量，無法再通過關閉、減少伺服器數量節省能源。

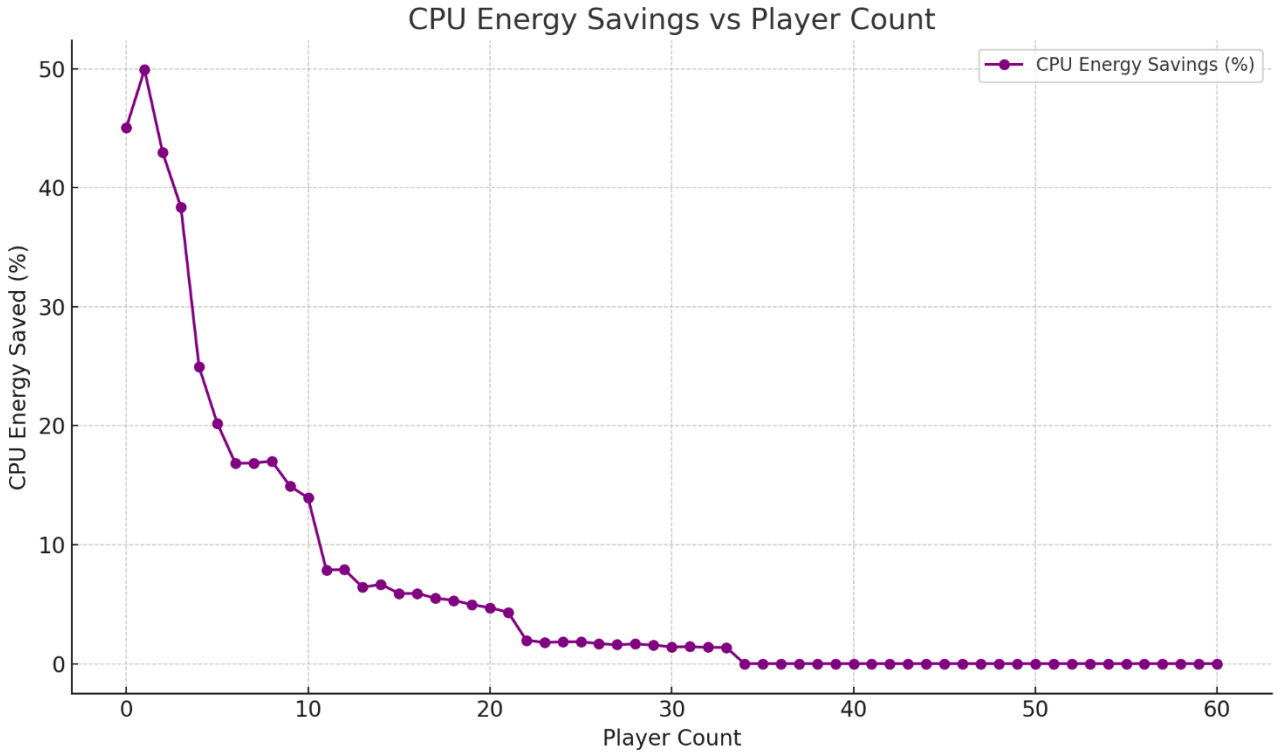
表 2

玩家人數對應節能效率

player_count	cpu_energy_saved
0	45.04359
1	49.90738
2	42.94181
3	38.37911
4	24.94891
5	20.2045
6	16.83312
7	16.86613
8	17.02048
9	14.93557
10	13.91962
11	7.882659
12	7.904379
13	6.412716
14	6.675569
15	5.89922
16	5.909802
17	5.508754
18	5.31409
19	4.973286
20	4.697688
21	4.329813
22	1.970677
23	1.786065
24	1.84867
25	1.844521
26	1.681663
27	1.595635
28	1.667649
29	1.559448
30	1.408227
31	1.433004
32	1.375289
33	1.3692
34	0.79243

圖 12

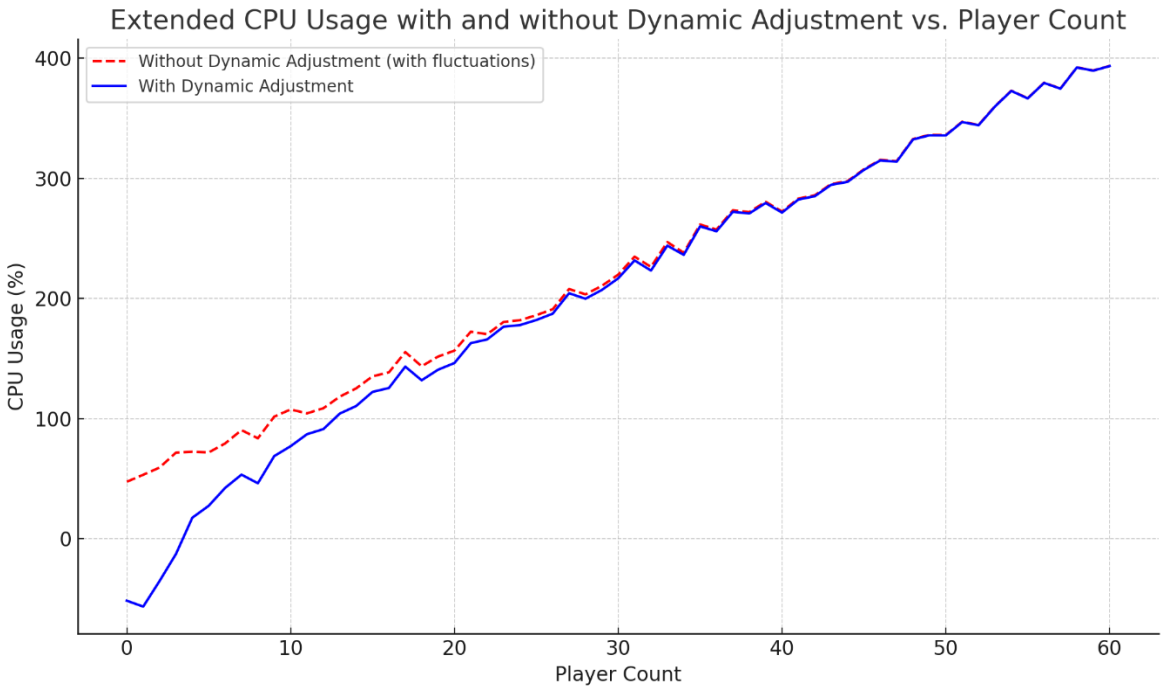
CPU 節能與玩家人數關係折線圖



註：由作者製作

圖 13

動態調節與未調節伺服器數量對 CPU 使用率與玩家數量關係的比較



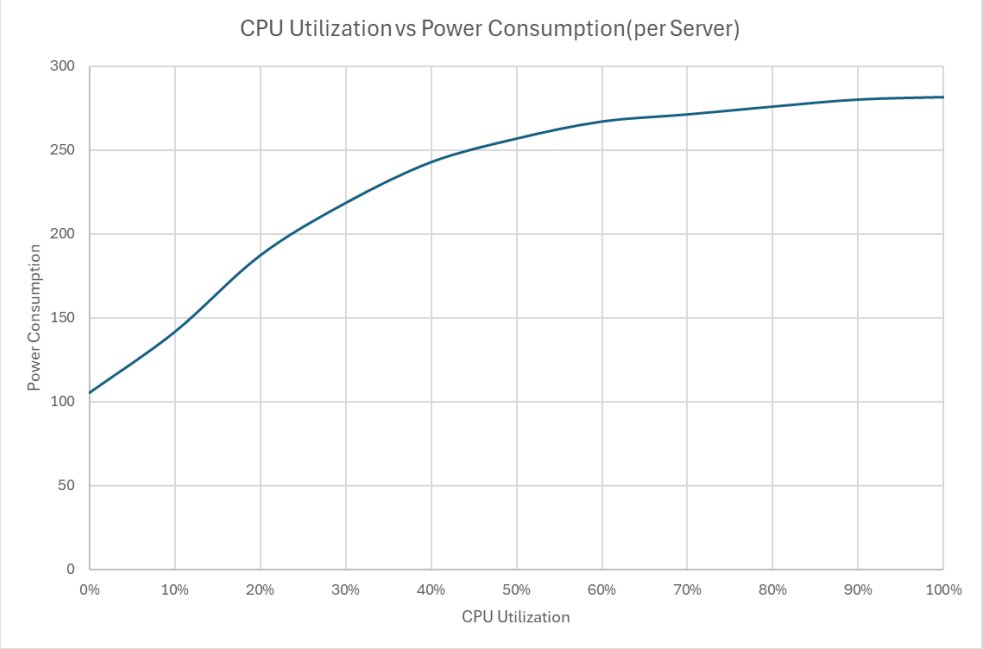
註：由作者製作

三、使用動態調整後節省的電能

透過圖14與圖15可以觀察到，隨著CPU使用率的增加，伺服器的功耗也會增加，呈現非線性關係。在低附載時，功耗增加的幅度較高負載時明顯，當負載接近極限時，功耗的增加趨於平緩。綜合圖14、圖15與前述圖表，我們可以發現，雖然整體還是隨著人數增加效率降低，但當玩家人數介於8到30人之間時，伺服器的實際節能效率比僅參考CPU使用率的節能效率數據更高。

圖 14

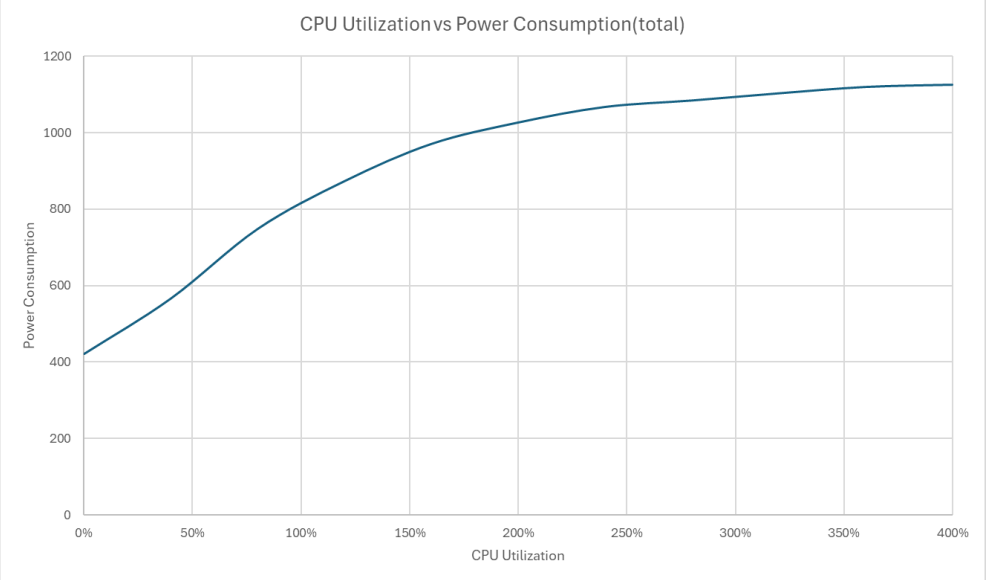
CPU 使用率與功耗之關係(一台伺服器)



註：由作者製作

圖 15

CPU 使用率與功耗之關係(所有伺服器)



註：由作者製作

四、玩家遊玩行為影響伺服器硬體負載

透過擴充套件以及分析系統，我們發現有一些玩家行為會對伺服器硬體使用效能有比較明顯的影響。在下列兩個公式中(參見公式一與公式二)，所有玩家行為事件是以「每分鐘發生的次數」為單位來計算的權重。另外從表三可以發現，玩家總數對伺服器負載影響最大，對 CPU 和 RAM 的權重分別為 1.09 和 0.58，表示隨著玩家數量增加，伺服器資源使用也會跟著上升。區塊載入和實體總數量(包含生物及怪物)也有較高影響，特別是當同時有大量實體在同一區域時，不只TPS會明顯下降造成卡頓，CPU 和 RAM 使用率也會跟著上升。另外，方塊更新(破壞或放置)、物品互動(撿起掉落物...等)、玩家互動等行為同樣會造成影響，不過相對比較不明顯。基於這些權重，本研究針對這次使用的主機配置建立了 CPU 和 RAM 使用率的模型。這些模型為理解並量化玩家行為對伺服器資源消耗的影響提供了依據，並可用於動態調整伺服器資源配置，以達到高效的伺服器管理和節能目標。

公式一：伺服器CPU使用率預測模型

$$\begin{aligned} cpu\_usage = {} & 1.09 \times Total\ Player + 0.35 \times Chuncks\ Load \\ & + 0.55 \times Total\ Entity + 0.57 \times Block\ Update\ \&\ Interaction \\ & + 0.39 \times Item\ Interaction + 0.24 \times Player\ Interaction \\ & + 0.4 \times Portal\ Interaction + 0.19 \times TNT\ Activation \\ & + 0.42 \times Crafting\ and\ Enchanting \\ & + 0.65 \times Chest\ and\ Furnace\ Interaction + 5.0 \end{aligned}$$

公式二：伺服器RAM使用率預測模型

$$\begin{aligned} ram\_usage = {} & 0.58 \times Total\ Player + 0.15 \times Chuncks\ Load \\ & + 0.32 \times Total\ Entity + 0.45 \times Block\ Update\ \&\ Interaction \\ & + 0.36 \times Item\ Interaction + 0.25 \times Player\ Interaction \\ & + 0.22 \times Portal\ Interaction + 0.27 \times TNT\ Activation \\ & + 0.22 \times Crafting\ and\ Enchanting \\ & + 0.48 \times Chest\ and\ Furnace\ Interaction + 100 \end{aligned}$$

表 3

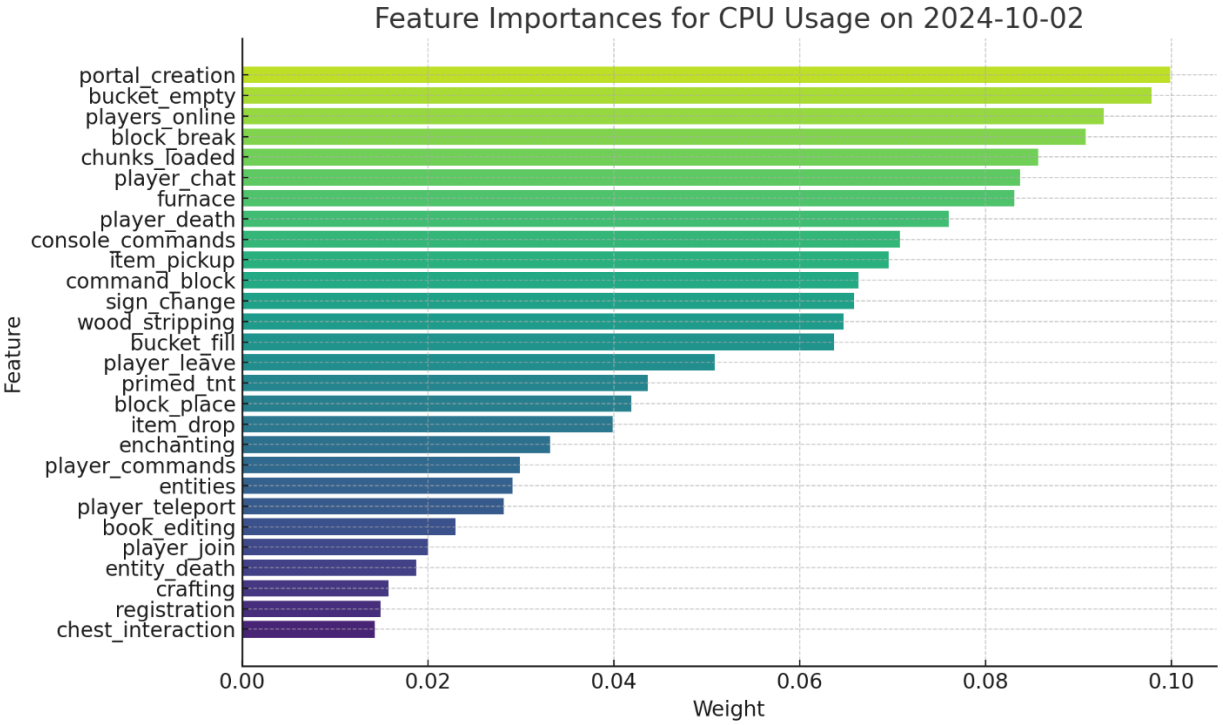
玩家事件與各項硬體數據影響對照表

事件在一分鐘內發生次數	CPU 權重	RAM 權重
Total Player	1.09	0.58
Chunks Load	0.35	0.15
Total Entity	0.55	0.32
Block Update & Interaction	0.57	0.45
Item Interaction	0.39	0.36
Player Interaction	0.24	0.25
Portal Interaction	0.4	0.22
TNT Activation	0.18	0.27
Crafting and Enchanting	0.42	0.22
Chest and Furnace Interaction	0.65	0.48

註：由作者製作

圖 16

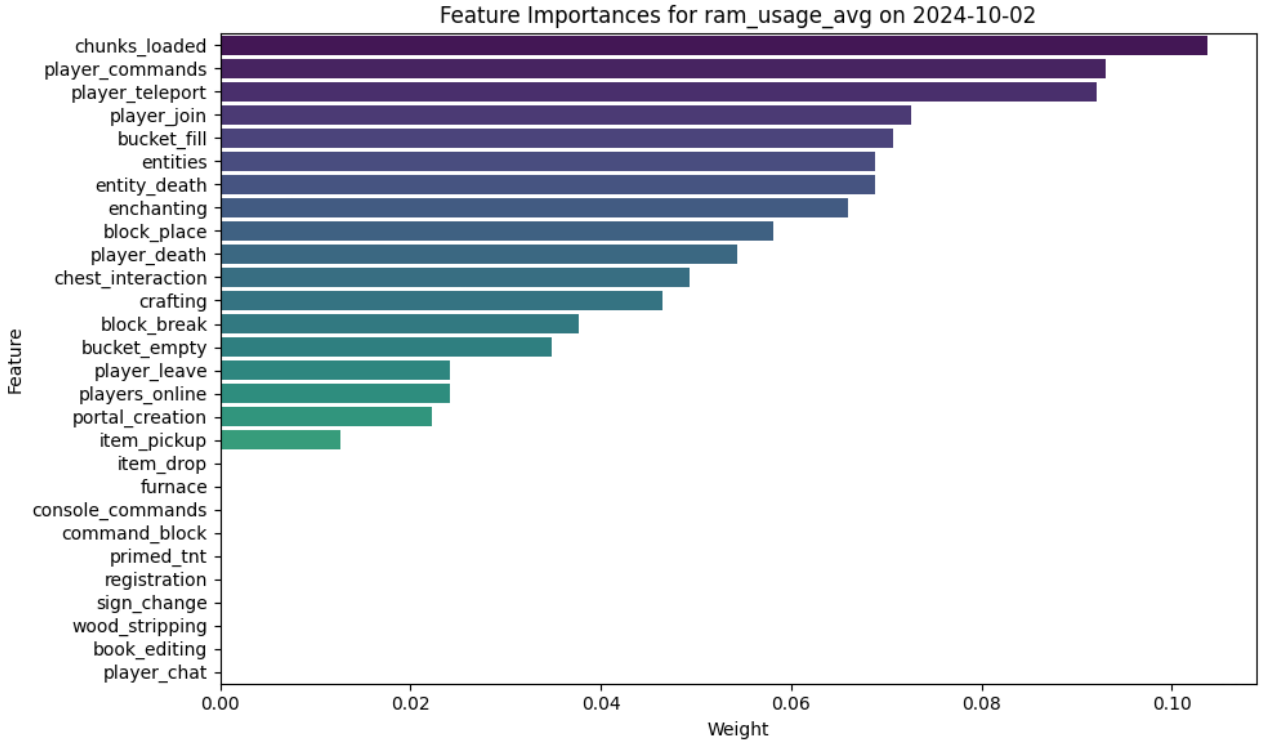
各個事件影響cpu使用率長條圖



註：由作者製作，x軸為每分鐘事件發生次數對硬體影響的權重，y軸為事件名稱

圖 17

各7事件影響記憶體長條圖



註：由作者製作，x軸為每分鐘事件發生次數對硬體影響的權重，y軸為事件名稱

研究結果顯示，隨著玩家數量增加，伺服器的CPU和RAM使用率顯著上升。透過疊加的時間序列圖可以知道，本次研究使用的伺服器在玩家數達到12~15人時會達到性能上限，因此當人數接近或超過的時候需要動態增加伺服器數量來分擔負載，以避免資源過載造成遊戲體驗下降。另外，本研究透過玩家數量與CPU/RAM使用率之間的多重迴歸模型，並使用加權平均法來估算伺服器效能狀況。

為了達到動態調節的效果，本研究使用Spigot和Bungeecord API開發動態調節擴充套件，使伺服器數量能根據負載情況實現自動啟動或關閉。在伺服器處於低負載情境下，動態調節系統會自動關閉所有空閒伺服器，並發現在玩家人數低於15人之前節能效率最為顯著，每個伺服器平均能省10~12%的CPU使用率。然而，當玩家數量增加，節能效果會越來越不明顯。當玩家數達到20人以上時，系統的節能效率降到只剩約5%；而在35人以上的情況下，已經沒有空閒的伺服器可以關閉，因此動態調節的節能效果在中低負載較為明顯。

此外，本次研究還分析了不同玩家行為對伺服器對CPU和RAM的負載影響。結果顯示，玩家數量對伺服器資源消耗的影響最大，其次是區塊載入、實體生成（如怪物和動物）及方塊互動等行為。在這些高影響事件的情境下，伺服器的資源消耗會顯著增加。

柒、結論與未來展望

一、結論

本研究透過動態調節伺服器數量，提供提升Minecraft伺服器的能源使用效率的辦法。在伺服器資源與玩家數量之間的關聯性分析中，我們發現隨著玩家數量的增加，伺服器的 CPU 和 RAM 使用率也顯著上升。在低負載時透過動態減少伺服器數量，能有效降低資源消耗達到節能效果，而在高負載的情境下，節能效果會隨伺服器達到運行上限而減弱。整體而言，動態調節機制成功提升了伺服器運行效能效率，並提供了節能的具體數據參考。

(一) 動態調節機制能明顯降低中低負載時的 CPU 和 RAM 使用率。

(二) 玩家數量與伺服器硬體使用率有正相關，玩家數量增加會導致伺服器硬體資源使用上升。

(三) 在伺服器達到高負載時，節能效果較為有限。

（四）開發的動態調節擴充套件讓伺服器自動管理更簡單。

二、未來展望

本計畫預計在這次研究的基礎上使用更進階的機器學習技術，透過提高伺服器負載預測的精準度，並對伺服器資源的動態分配進行優化，能收到更多數據提高精準度。由於伺服器類型的不同以及玩家類型不同，未來也希望能設計更適合所有類型伺服器的資源調節策略。

（一）希望使用更進階的機器學習技巧，以增強系統預測能力。

（二）將動態調節機制應用於更多不同平台。

（三）探討針對不同類型遊戲行為的資源調整機制。

（四）使用更多的數據提高系統準確度。

捌、參考文獻

一、中文部分

Enago Academy. (n.d.). 概念研究與實證研究. 取自 https://www.enago.tw/academy/%e6%a6%82%e5%bf%b5%e7%a0%94%e7%a9%b6%e8%88%87%e5%af%a6%e8%ad%89%e7%a0%94%e7%a9%b6/

groots.(2019, October 26). 數據分析師必備的 10 大技能，大多數人只知道一半. iT 邦幫忙.取自https://ithelp.ithome.com.tw/articles/10221596

Jason. (2023). 獨立樣本 t 檢定-理論. Medium. 取自 https://medium.com/@jason8410271027/%E5%AD%B8%E7%BF%92%E7%AD%86% E8%A8%98-%E7%8D%A8%E7%AB%8B%E6%A8%A3%E6%9C%ACt%E6%AA%A2%E5%AE%9A-%E7%90%86%E8%AB%96-python%E5%AF%A6%E4%BD%9C-146699cf0bd9

Minecraft 官方網站. (n.d.). Minecraft 關於. 取自 https://help.minecraft.net/hc/en-us

Mojang Studios. (n.d.). Minecraft Java 版與基岩版一體化. 取自 https://www.minecraft.net/zh-hans/store/minecraft-java-bedrock-edition-pc

李筠茱（2019）。臺灣教師使用歐盟Go-Lab 系統進行線上探究式教學之推展研究。〔碩士論文。國立臺灣師範大學〕臺灣博碩士論文知識加值系統。https://hdl.handle.net/11296/54c3w4。

張茵婷（2018）。程式設計課程融入體驗學習之探究。〔碩士論文。國立臺灣師範大學〕臺灣博碩士論文知識加值系統。https://hdl.handle.net/11296/fz42u3。

楊雅雯（2017）。玩中學－數位遊戲式學習. 臺灣教育評論月刊, *6*(9), 300–302. http://www.ater.org.tw/journal/article/6-9/free/22.pdf

二、 西文部分

About Spigot. (2021, Winter 4). Spigotmc. https://www.spigotmc.org/wiki/about-spigot/

Arora, S. K., & Shah, D. (2016). Writing Methods: how to write what you did?.Indian pediatrics, 53, 335-340.

Brooke, J. (1996). SUS: A quick and dirty usability scale. Retrieved From https://www.researchgate.net/publication/228593520_SUS_A_quick_and_dirty_usability_scale

Alan, I., Arslan, E., & Kosar, T. (2014). Energy-aware data transfer tuning. *Proceedings of the 14th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing (CCGrid 2014)*, 626-634. https://doi.org/10.1109/CCGrid.2014.117

Alan, A., et al. (2014). SCP, rsync, FTP, bbcp, and GridFTP data transfer tools. *Journal of Data Transfer and Networking*, 25(2), 150-162.

Bebbington, S. (2014). Minecraft as a creat77ive tool. *International Journal of Game-Based Learning (IJGBL), 4*(2), 1-14. https://doi.org/10.4018/ijgbl.2014040101

Bohra, A. E. H., & Chaudhary, V. (2010). VMeter: Power modelling for virtualized clouds. *Proceedings of the International Conference on Parallel Processing Workshops*, 1-8. https://doi.org/10.1109/ICPPW.2010.76

Bukkit. (n.d.). Retrieved from http://bukkit.org/

BisectHosting. (n.d.). How to choose which Minecraft server to install: Vanilla, Spigot or Paper. Retrieved from https://www.bisecthosting.com/clients/index.php?rp=/knowledgebase/253/How- to-choose-which-Minecraft-server-to-install-Vanilla-SpigotorPaper-Modded.html

Game Features. (n.d.). Minecraft Education. https://educommunity.minecraft.net/hc/en-us/articles/360047117692-FAQ- Game-Features

Gupta, A. (2021). Ridge regression and its application in server performance models. *International Journal of Data Science*, 25(6), 98-115.

Hochreiter, S., & Schmidhuber, J. (1997). Long short-term memory. *Neural Computation*, 9(8), 1735-1780.

Hitchings, H. (n.d.). Minecraft mods: A beginner's guide with tips explained. The Gamer. Retrieved from https://www.thegamer.com/minecraft-mods-beginners- guide-tips-explained/

Kansal, A., et al. (2010). Evaluation of SPECcpu 2006 and IOmeter for server performance analysis. *Journal of Server Performance*, 22(1), 89-102.

Lischka, A. E.7 (2015). Addressing needs of diverse learners with apps for learning mathematics. *Teaching Children Mathematics*, *21*(1), 56-59. https://doi.org/10.5951/teacchilmath.21.1.0056

Lee, T., & Park, C. (2020). Player behavior analysis and server load prediction using regression models. ACM Transactions on Gaming Analytics, 20(2), 123-138.

Saiful Bahry, F. D., Masrom, M., & Masrek, M. N. (2021). Measuring validity and reliability of website credibility factors in influencing user engagement questionnaire. International Journal of Web Information Systems, 17(1), 18-28.

Smith, M. (n.d.). Minecraft Server Types. Shockbyte. Retrieved March 31, 2024, from https://shockbyte.com/billing/knowledgebase/32/Minecraft-Server- Types.html

Smith, J., & Johnson, A. (2021). Ethical Considerations and Practical Guidelines for Participant Recruitment in Research Studies. *Journal of Ethics in Research*, 35(1), 112-127.

Smith, J. (2020). Lasso regression for server load forecasting. *IEEE Transactions on Cloud Computing*, 8(4), 1127-1135.

SpigotMC. (n.d.). Home. Retrieved from https://www.spigotmc.org/
SpigotMC. (n.d.). Logger 1.7-1.20. Retrieved from https://www.spigotmc.org/resources/logger-1-7-1-20.94236/
Monteclaro, A. J. (2023, September 15). What is a server? ServerWatch. Retrieved from https://www.serverwatch.com/guides/what-is-a-server/
Oracle. (n.d.). What is a database? Retrieved from https://www.oracle.com/tw/database/what-is-database/
Pedram, M., & Hwang, I. (2010). Power and performance modeling in a virtualized server system. *Proceedings of the International Conference on Parallel Processing Workshops*, 520-526. https://doi.org/10.1109/ICPPW.2010.76
Rockstar Games. (n.d.). GTA Online 指南. Retrieved from https://www.rockstargames.com/tw/gta-online/guides/7772?section=8oo4
Rouhiainen, A. (2023). Teaching Boolean algebra and logic gates to students from junior high school. Theseus. Retrieved July 30, 2024, from https://www.theseus.fi/bitstream/handle/10024/861690/Aaro_Rouhiainen_Thesis.pdf?sequence=2
Redbeet Interactive. (n.d.). *Raft.* Retrieved October 17, 2024, from https://raft-game.com/
Wang, S., et al. (2019). Handling multicollinearity in server performance prediction models. *Journal of Data Science and Analytics*, 17(1), 54-72.

【評語】190004

1. 本作品設計一個可以動態調整多個網路伺服器服務平台的機制，以節省ＣＰＵ資源，透過 Bungeecord 讓五台主機分別運行並互相連通，利用迴歸分析預測CPU與ＲＡＭ使用率與玩家行為變數間的關係，實驗系統平台尚完整。

2. 建議可以加上玩家滿意度分析，是否因動態調整而影響遊戲平台反應時間。

3. 建議可比較多種常見的方法用於節能動調整，以及使用現有 AI 與深度學習模型，來改善節能效率。