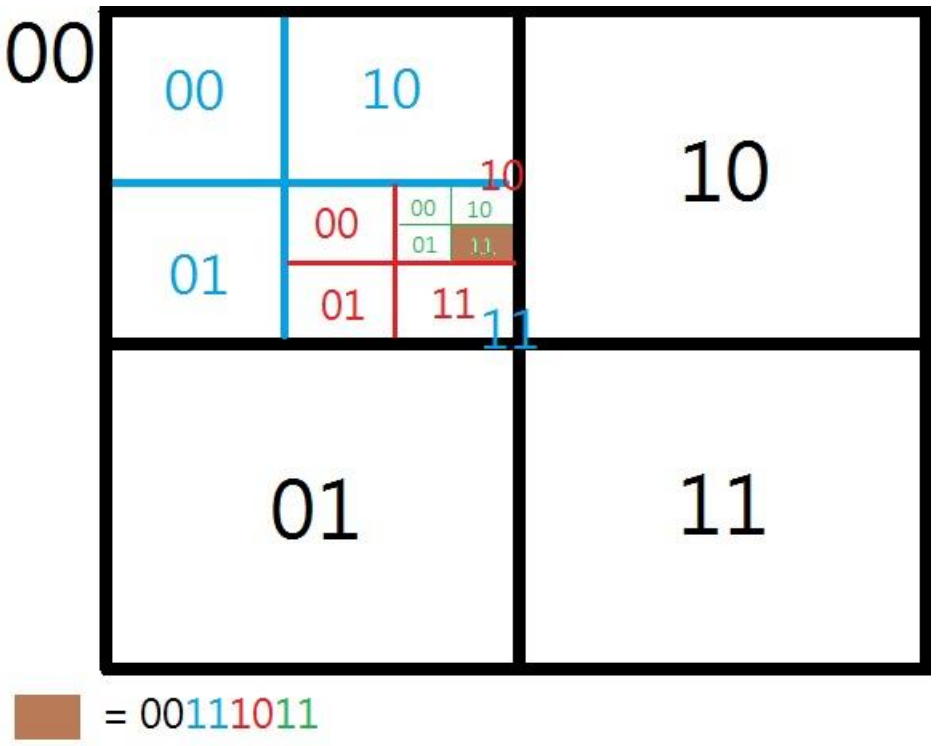
第三節 Geohash 演算法於本專案之探討與應用

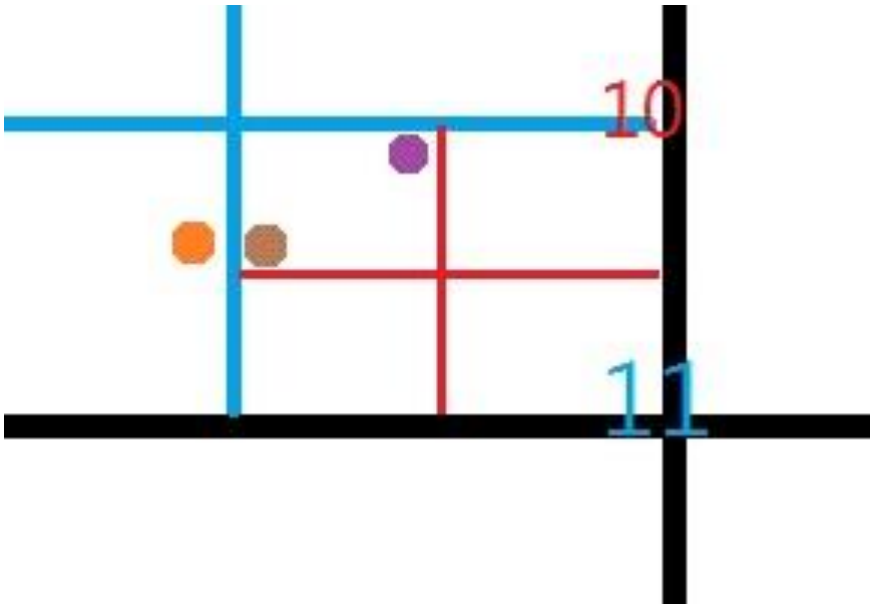
5.3.1 Geohash 原理探討

Geohash 是將區域切割成 N 個區域，並且再將每一個區域遞迴持續切割成 N 個區域，直到最小區域逼近到指定精確度的一種空間標記法，為四叉樹的一種延伸應用方式，使搜尋目標可達到 O(log(N))的時間複雜度。最後產生一個 hash code，即為目標所在位址範圍。下圖由 N=4，並且使用二進位標記法做舉例。

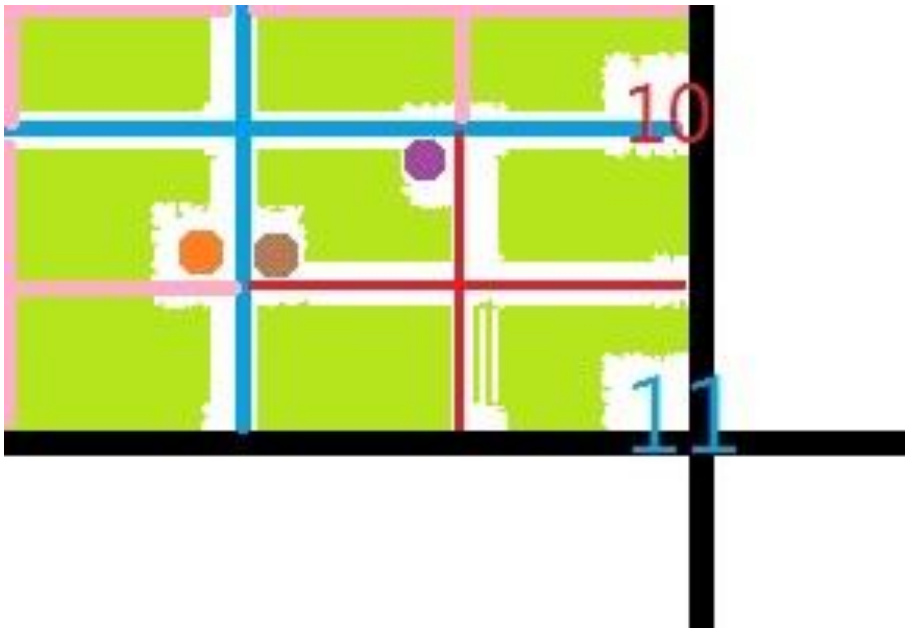


5.3.2 Geohash 的限制與解決之道

問題：若單純以「同個格子內的點就是最近的點」的想法出發，「咖啡色點」明顯與與「橘色點」較近，但是「咖啡色點」與「紫色點」位於同一區，所以最後「最近點」會抓到「紫色點」。(局部放大上圖)



解決之道：使用「查表法」抓取周圍八個方格，共九格。取得九個方格中所有點的位置再進行「最近點計算」。



5.3.3 Geohash 查表法取得鄰近八格區域

文獻：https://github.com/davetroy/geohash-js
探討 GitHub 上的資源，使用 javascript 實作的查表法，並於 php 語言上實作
其他輔助資源：http://www.itdadao.com/articles/c15a132346p0.html

以一組長度 20bit 的 hash code 為例：11010 11100 00101 01010
定義 base32 編碼從小到大順序：0123456789abcdefghijklmnopqrstuv
定義起始 bit 使用經度開始切割
使用 base32 編碼為：qs5a，定義 q 為第一位，s 為第二位…類推

將 geo hash code 編碼成 base32 之後可以將 geohash 視為：在 32 格中尋找某一格，再將這格切割成 32 格。於本例中重複此動作 4 次可符合。

觀察發現，base32 編碼的「奇數位」是按照「經緯經緯經」切割；「偶數位」按照「緯經緯經緯」切割，故需產生兩組表格對應「奇數位」與「偶數位」。

查表尋找鄰近八格

如何尋找編號 qs5a 格子鄰近的八格？可透過下表：

odd:

0	1	4	5	g	h	k	l
2	3	6	7	i	j	m	n
8	9	c	d	o	p	s	t
a	b	e	f	q	r	u	v

==========

even:

v	t	n	l
u	s	m	k
r	p	j	h
q	o	i	g
f	d	7	5
e	c	6	4
b	9	3	1
a	8	2	0

「qs5a」最後一碼為「偶數位」，觀察「even」表格，「a」的鄰近八格為b、9、8、1、0、v、t、l。其中「b、9、8」並未跨界，「1、0、v、t、l」跨界。故可確定八格中有三格為「qs5b」、「qs59」、「qs58」。

「1、0」向右超界；「v、t」向下超界；「l」向左下超界。超界者須觀察前一位，也就是「5」。「5」為「奇數位」，觀察「odd」表格。「1、0」向右超界，「5」應修正為「g」；「v、t」向下超界，「5」應修正為「7」；「l」向左下超界，「5」應修正為「6」。最後修正為「qsg1」、「qsg0」、「qs7v」、「qs7t」、「qs6l」。若持續到最高位數仍超界，代表此位置超出 Geohash Map 所設定的支援範圍。

最後得到周圍八格之編碼為：「qs5b」、「qs59」、「qs58」、「qsg1」、「qsg0」、「qs7v」、「qs7t」、「qs6l」。

表格的產生

「odd」或「even」表格，共需產生 32 個唯一的編碼，各編碼尋找位置如下說明，以「odd」表格為例：

odd: 5 = 00101

0	1	4	5	g	h	k	l
2	3	6	7	i	j	m	n
8	9	c	d	o	p	s	t
a	b	e	f	q	r	u	v

以「5」為例，二進位表示為「00101」。根據 Geohash 左邊為 0、右邊為 1；上面為 0、下面為 1 的規則，可找到「5」應放置的位置。

「odd」表格較「even」表格，使用經度多切割一次，故寬度應為兩倍；反之「even」使用緯度多切割一次，故長應為兩倍。

5.3.4 Geohash 於本專案之應用

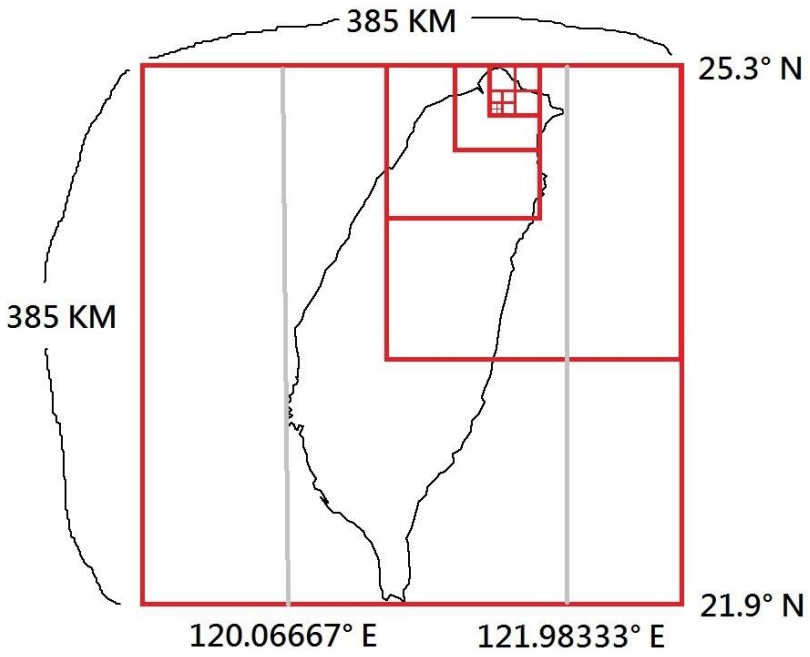
Geohash 是將某一限定範圍區域做巢狀切割的技術，因此必須人工設定邊界範圍，以及巢狀切割的次數，使結果符合實際需求。

設定邊界範圍須符合實際地理情況，本次專案需要範圍涵蓋全台灣。台灣為一南北狹長型的區域，故選擇台灣由極北至極南的長度作為 Geohash 正方形的邊長。至於正方形的理由是因為 Geohash 處理正方形的情況較單純好處理。選擇切割的次數為 10 次，理由則在下方推算。

台灣南北總長為 385km=385000m，使用一個長為 385000m 的正方形框住台灣，並且使用 2x2=4 格的 Geohash，遞迴切割 10 次可得到最小的正方形長為 385000/(2^10)m 大約為 376m。

以店家所在方格為中心，取鄰近八格，共九格形成一正方型區域，邊長為 376*3=1128m。以人口密度最大的台北市 9,939.84 人/km(資料取自 wiki)換算，9,939.84*1.128=11212.14 人/km。也就是說在常理之下，位於此九宮格範圍內的人數最多為 11,212 人，伺服器在一次的 request 中比較次數最多大約 1 萬次，屬於合理範圍。當然不可能這 1 萬多人同時都是送貨員，此為極限狀態下的推算，實際情況應小於此極限值。

使用二進位表示方式，在二維座標中切割 10 次，所以 hash code 的長度為 10*2=20bits(x 軸 10 次，y 軸 10 次)。以下僅為示意圖：



5.3.5 實作工具

使用 php 語言配合 Mongo DB 實作此功能。php 作為本次伺服器實作語言，因此相容性最好。Mongo DB 作為樹狀結構的 No SQL 資料庫，對於每個「Geohash 格子」內有 0 個以上「不定數量」的「送貨員節點」，這種資料格式相性很好。使用 Composer 作為 php 與 Mongo DB Shell 溝通的中介元件。