submit for poster session

σ-N 天際線: 一種新的天際線搜尋類型

(σ-N Skyline: A new type of skyline search)

黃璽合
國立成功大學資訊工程所
kid863@hotmail.com

李 強
國立成功大學資訊工程所
leec@mail.ncku.edu.tw

摘要

天際線查詢問題在近年來引起相當大的關注。給定一個資料集，天際線查詢會回傳那些不被其他點所支配的資料點。在這篇研究中，我們將天際線查詢擴展到 σ-neighborhood 天際線查詢(σ-N 天際線查詢)。相較於過去的天際線查詢，σ-N 天際線查詢不僅僅找出天際線點，也會擷取與天際線點競爭的資料點。由於σ-N 天際線查詢可以提供更有彈性的答案給使用者，因此應用在策略抉擇、市場分析與商業規劃上都很有幫助。

本篇論文定義了一個全新的問題，提出一個新的索引樹與演算法來解決，並透過模擬實驗來展示該演算法之有效性與效率。

關鍵詞：資料庫、查詢處理、天際線查詢、索引樹。

Abstract

The skyline query problem has attracted considerable attention in recent years. Given a dataset, a skyline query returns data points that are not dominated by other points. This study extends the concept of skyline query to a so-called *σ*-neighborhood skyline query (*σ*-N skyline query). In contrast to the past skyline query, the *σ*-N skyline query not only finds the skyline points, but also retrieves points that are competitive with the skyline points. The *σ*-N skyline query can be useful for applications such as decision making, market analysis, and business planning, as it is able to provide more flexible answers for a user. This paper defines the novel problem, proposes a new index tree and intelligent algorithms to resolve this problem, and conducts a set of simulations to demonstrate the effectiveness and efficiency of the algorithms.

Keywords: *Database, Query processing, Skyline query, Index tree.*

一、簡介

近年來，天際線搜尋由於其在 multi-criteria decision making 的廣泛應用，因此其被認為是資料庫最關鍵的技術之一。對於一個包含有資料點 $p_1, p_2, \ldots, p_l$,的資料集 D，天際線搜尋將會回傳所有不被支配的資料點。其中，如果一個資料點 p_j 在所有維度中都不比 p_i 差，且至少有一個維度比 p_i 好，則我們稱 p_j 支配 p_i。

不幸的，天際線搜尋有一個致命的缺點。如同我們所知的，天際線資料點是由一群不被任何人所支配的資料點所組合而成的。這也就是說，如果有一個資料點僅比其他資料點差一點，他也不會被回傳給使用者。這樣的定義可能違反大多數人的邏輯。使用者在選擇東西時，通常會為了選擇較好的物件而不介意多支付一些代價。因此我們需要創新一個能克服此限制的嶄新天際線定義。

為了解決此問題，我們在此論文中提出了 neighborhood-*σ* skyline (N-*σ* skyline)。我們將所有比天際線資料點 p_{sky} 差的資料點 p_i (i.e. dominated by p_{sky})，但卻落在一定範圍內的資料點稱為 N-*σ* skyline。其中，我們稱 *σ* 為距離的標準，並讓使用者定義之。在此，我們強調在此篇論文中所說到的距離，必須得要在所有維度都正規化到某一區間中，像是[0, 1] 或 [-1, 1]方能計算。而做正規劃的原因則是因為不同維度使用不同的單位，而我們必須消除此差異性。

二、設計方法、技術與步驟

過去的天際線演算法皆使用 R-tree 來作為發展的基礎。儘管如此，R-tree 卻無法滿足我們解決 N-*σ* skyline 問題的需求，因為 R-tree 無法避免 MBR 間大量的 overlap。這對 N-*σ* skyline 的查詢來說是一場災難。因為每個點通常都需要大量支配檢查與距離檢查，越多的點需要檢查代表我們需要花費更多的 cost 來完成。資料在 R-tree 中並不是用距離來作為分類

submit for poster session

的依據。這也就是說，在進行 N-σ skyline 的查詢時，MBR 不能提供給我們任何距離的資訊或特質。如果我們使用 R-tree 來做 index model 的話，我們將會另外需要大量的距離運算來找尋最終的結果，而這是一個非常耗費 cost 的過程。因此，為了克服 R-tree 在我們的 work 中所遇到的缺點，我們需要找尋另外的資料結構來有效的解決 N-σ skyline problem。

在此論文中，我們介紹 M^+-tree 來解決我們的問題，藉由此樹，我們可以大幅提昇我們處理查詢的表現，使用以「距離」為條件建構的 M^+-tree 將會比使用以「面積」為條件建構的 R-tree 具有幾項優勢。1)在目標的查詢中，使用 M^+-tree 將會比使用 R-tree 避免更多不必要資料處理與比較。2)由於 M^+-tree 是以距離為條件所建立的，節點裡面儲存了許多與距離相關的參數。因此在 σ-N skyline query 中可直接使用這些參數並節省許多查詢中會遇到的距離運算。而此基於 M^+-tree 處理 σ-N skyline query 的方法，稱為 Mσ-N algorithm。

Mσ-N Algorithm 是藉由三個 mechanisms 來提升 σ-N skyline query 的效率。σ-N skyline query 的效率主要是跟兩個參數 N_s 與 N_e 相關。N_s 代表每個 entry 所需進行空間重疊檢查的平均天際線數目。N_e 則代表查詢中所檢查過的 entry 數目。當 N_s 與 N_e 的數值越小時，σ-N skyline query 將能被操作的越快。此小節中前兩個 mechanism, inheritance-based pruning mechanism 與 summation checking mechanism 將用來降低 N_s。第三個 inequality-based pruning mechanisms 則用來降低 N_e。

(一) Inheritance-based pruning mechanism

Inheritance-based pruning mechanism 在 Mσ-N Algorithm 中是被用做子程式「Examination between the entries in Heap and the σNRs in *LS*」的前置過濾器，此 mechanism 能透過過往檢查過的 MMBR 與 σNR 的空間關係來快速推測 MMBR 子節點與某些 σNR 的空間關係。若一個 MMBR 的子節點能透過此 mechanism 直接獲得他與某些 σNR 的空間關係，則此 MMBR 的子節點與這些 σNR 的關係將不再需要子程式「Examination between the entries in Heap and the σNRs in *LS*」來做檢查。也因此查詢中 N_s 的數目將被降低並提升查詢的速度。

(二) Summation checking mechanism

Summation checking mechanism 則是被用做子程式「Examination between the entries in Heap and the σNRs in *LS*」的加速器，能加速的地方包含兩部分：1)在執行此子程式前事先過濾掉不需要檢查的 entry 與 σNR 組合。如此，每個 entry 僅需與較少的天際線資料點進行空間重疊關係檢查即能得知該進行何種處理。亦即，σ-N skyline query 所需的 N_s 數目降低。2)由於 summation checking mechanisms 的運算方式將不會受到維度的高低而影響。因此我們也可避免在處理高維資料時，此子程式效率降低的情況。

(三) Inequality-based pruning mechnisms

Inequality-based pruning mechnisms 是當一個 MMBR 是 partially contained in some σNRs 時，要把其內的所有的子節點插入 heap 時所使用的，會針對此狀況做討論，是因為此處為 entry 增加的主要原因之一。若我們能從此處降低要插入 heap 的子節點數目，則我們可以提升整體演算法的效率。此外，由於 MMBR 內可能為 point 或 MMBR，所以本 mechanism 也可分為兩個部分來討論。假設 MMBR 內的子節點為 point，則本 mechanism 可快速檢查這些 point 是否為 σ-N skyline point。若是 σ-N skyline point，則直接將子節點放入 σ-N skyline point 的結果，而不用插入 heap 中。若不是 σ-N skyline point，則不插入 heap 中進行更進一步的處理。只有當本 mechanism 無法判斷該 point 為 σ-N skyline point 或不為 σ-N skyline point 時，該 point 才會插入 heap 中做更進一步的處理。相似的，假設 MMBR 內的子節點為其他的 MMBR。則本 mechanism 能快速判斷這些 MMBR 內的資料點是否全為 σ-N skyline point 或全不為 σ-N skyline point。若其內部的資料點全為 σ-N skyline point，則把這些資料點全部插入 σ-N skyline point 的結果中。若內部的資料點全不為 σ-N skyline point，則這些資料點將在演算法中被忽略。只有當 mechanism 無法判斷子 MMBR 內資料點的狀況或是子 MMBR 內僅有部分資料點為 σ-N skyline point 時才需要把子 MMBR 插入 heap 中做處理。

三、實驗模擬

submit for poster session

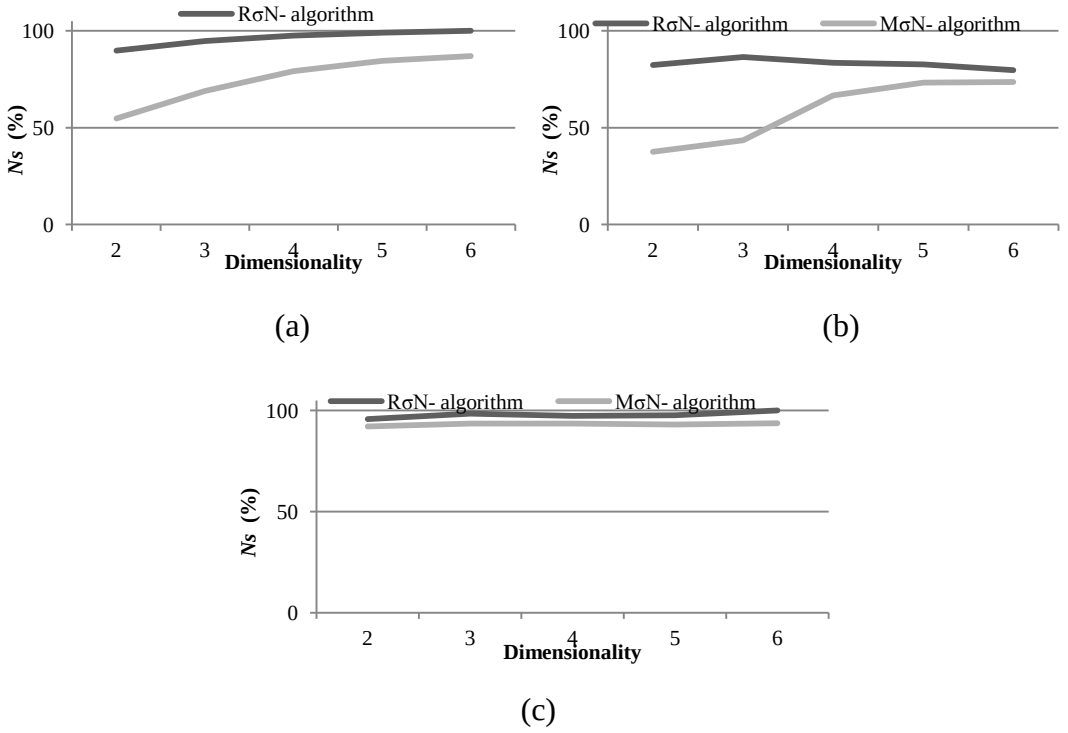

Figure 1. The result of N_S and considering the effect of dimensionality (a) independent dataset. (b) anti-correlated dataset. (c)NBA player dataset.

我們使用一些實驗來驗證目標方法的有效性。本實驗使用 independent dataset，anti-correlated dataset 以及 NBA 球員 dataset 來做驗證。這三個 dataset 是最常被用來檢驗 skyline search algorithm 的有效性。在本模擬中，每個 dataset 每個資料點的維度資料值都將被正規化到[0,1]，三個 dataset (以 2 維為例)在正規化到後[0,1]。三個 dataset 的維度值都將在 2 至 6 維間變動。independent dataset，anti-correlated dataset 的 cardinality 為 1M，NBA player 的 cardinality 則為 20788。此外，independent dataset 及 anti-correlated dataset 的 σ 值會在 0.01 到 0.1 之間變動，但 NBA 球員 dataset 的 σ 值會在 0.15 到 0.25 之間變動。這是因為 NBA 球員 dataset 的 skyline point 與其他資料點間的距離較大。所以我們必須將 σ 值設的大一點才會查詢到較多的 σ-N skyline point。此外，simulation 中所有的測試結果皆為 30 次實驗平均的結果。本章的程式是以 MATLAB®開發，模擬電腦則使用 2.93GHz 的 Intel Core2 Duo CPU 並配合上 2GB 的記憶體，作業系統則為 Microsoft Windows XP。

我們固定 σ 值(independent dataset，anti-correlated dataset 為 0.05，NBA 球員 dataset 為 0.2)並考慮 2 維到 6 維狀況下演算法的效能。結果分成以下幾個部分討論，包含 σ-N skyline point 的數目、N_s 的數目、N_e 的數目與處理時間。

Figure1. 為 N_s 的結果，其中 Figure 1.(a)為 independent dataset 的結果， Figure 1.(b)為 anti-correlated dataset 的結果，Figure 1.(c)則為 NBA player dataset 的結果。在 Figure 1.(a)與 Figure 1.(b)大部分的情況下，我們皆可看到 Mσ-N algorithm 的 N_s 會比 Rσ-N algorithm 的少了約 20 到 40 個百分比。而這也意味著 Mσ-N algorithm 中 inheritance-based pruning mechanism 與 summation checking mechanism 的有效性。此外，Rσ-N algorithm 的 N_s 在 independent dataset 的 case 中約等於 100%（如 Figure 1.(a)所示），而在 anti-correlated dataset 中則約等於 80%（如 Figure 1.(b)所示）。這代表了在 Rσ-N algorithm 中，每個 entry(i.e., a RMBR or a point)需要與接近全部的 skyline point 做檢查。Mσ-N algorithm 的 N_s 則不會維持固定，他會隨著維度的增加而增加（如 Figure 1.(a)及 Figure 1.(b)所示）。這是因為 Mσ-N algorithm 中 summation checking mechanism 的效能會隨著維度增加而稍微變差。當維度變高，具有相同 summation 值的資料點也就越多，所以 mechanism 能過濾掉不需檢查的 σNR 就越少(i.e., N_s 會變多)。最後，在 Figure 1.(a)與 Figure 1.(b)中我們也可看到 independent dataset 的 N_s 會比 anti-correlated

dataset 高。這是因為 anti-correlated dataset 的 *σNR* 容易有多個重疊在一起的現象。也因此一個 *σ*-N skyline point 可能同時附屬於多個天際線資料點上。這代表 entry 在與每一個 *σNR* 檢查時，有較高的機會會成為或包含 *σ*-N skyline point。一旦這個狀況成立，該 entry 就不用再與任何 *σNR* 進行檢查。亦即，N_s 會下降。這也就是說，anti-correlated dataset 的 N_s 會較 independent dataset 為少。最後，在 Figure 1.(c) 中，M*σ*-N algorithm 的 N_s 只有比 R*σ*-N algorithm 的少了約 5 個百分比。這是因為 NBA player dataset 的 skyline point 之總和在正規化到[0, 1]後幾乎都相等。所以 M*σ*-N algorithm 中的 summation checking mechanism 的功用被降低了。

我們皆可看到執行 M*σ*-N algorithm 的時間比 R*σ*-N algorithm 快。這是因為在這兩個 dataset 中，M*σ*-N algorithm 的 N_s 及 N_e 皆比 R*σ*-N algorithm 的出色。又由於 *σ*-N skyline query 執行的效能是由 N_s 及 N_e 所決定的，所以 M*σ*-N algorithm 當然擁有較快的執行時間。至於 NBA player dataset，雖然 M*σ*-N algorithm 的 N_s 僅比 R*σ*-N algorithm 好一點點，但在 M*σ*-N algorithm 的 N_e 比 R*σ*-N algorithm 好上許多的情況下，M*σ*-N algorithm 的執行時間也會比 R*σ*-N algorithm 的快上許多。

四、致謝

This work was supported by NSC under the Grant NSC 100-2221-E-006-249-MY3.

參考文獻

[1] S. Borzsonyi, D. Kossmann, and K.Stocker, “The skyline operator,” in *Proc. International Conference on Data Engineering*, 2001, pp. 421-430.

[2] D. Papadias, Y. Tao, G. Fu, and B. Seeger, “An optimal and progressive algorithm for skyline queries,” in *Proc. ACM Special interest group on management of data*, 2003.