搜尋演算法

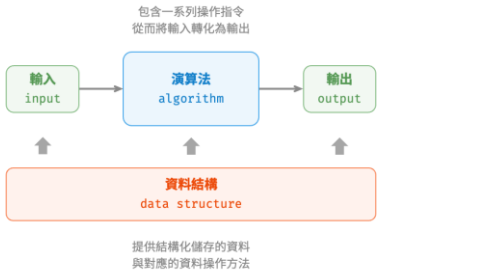
一、 先複習七年級時已學過的演算法定義及概念

- 演算法（algorithm）是在有限時間內解決特定問題的一組指令或操作步驟，它具有以下特性。
- 問題是明確的，包含清晰的輸入和輸出定義。
- 具有可行性，能夠在有限步驟、時間和記憶體空間下完成。
- 各步驟都有確定的含義，在相同的輸入和執行條件下，輸出始終相同

二、接著引導搜尋演算法

搜尋演算法是在資料集合中尋找特定資料的步驟式方法，有不同的演算法適用於不同資料結構和資料量，課本介紹了兩種演算法:基礎的線性搜尋法(適合無序資料)，另一個為需要事先排序資料的二分搜尋法。 配合最新時事, 補充 Google 搜尋演算法，AI 搜尋演算法。利用猜牌程式遊戲,引起學生興趣,並在遊戲中去體驗實作搜尋演算法的概念.(程式連結: https://scratch.mit.edu/projects/1079529374)

三、補充 （資料來源:https://www.hello-algo.com/zh-hant/chapter_searching/searching_algorithm_revisited/#1053）



給定大小為 n 的一組資料，我們可以使用線性搜尋、二分搜尋、樹查詢、雜湊查詢等多種方法從中搜索目標元素。各個方法的工作原理如圖 10-11 所示。

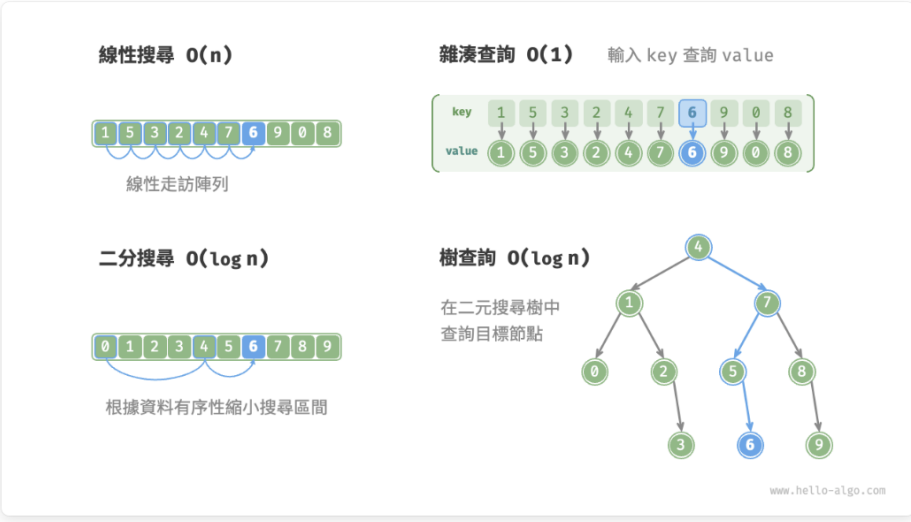


圖 10-11 多種搜尋策略

搜尋演算法的選擇還取決規模、搜尋效能要求、資料查詢與更新頻率等。

線性搜尋

通用性較好，無須任何資料預處理操作。假如我們僅需查詢一次資料，那麼其他三種方法的資料預處理的時間比線性搜尋的時間還要更長。

適用於體量較小的資料，此情況下時間複雜度對效率影響較小。

適用於資料更新頻率較高的場景，因為該方法不需要對資料進行任何額外維護。

二分搜尋

適用於大資料量的情況，效率表現穩定，最差時間複雜度為 。

資料量不能過大，因為儲存陣列需要連續的記憶體空間。

不適用於高頻增刪資料的場景，因為維護有序陣列的開銷較大。

雜湊查詢

適合對查詢效能要求很高的場景，平均時間複雜度為 。

不適合需要有序資料或範圍查詢的場景，因為雜湊表無法維護資料的有序性。

對雜湊函式和雜湊衝突處理策略的依賴性較高，具有較大的效能劣化風險。

不適合資料量過大的情況，因為雜湊表需要額外空間來最大程度地減少衝突，從而提供良好的查詢效能。