資料結構報告

41243117 吳承璿

10.23 2024

解題說明

1. 給定一個集合 S，我們對每個元素都可以選擇是否將其加入到當前子集中，這會產生兩個遞迴分支：
   - 不包含當前元素，直接進行下一層遞迴。
   - 包含當前元素，將其加入到子集中，再進行下一層遞迴。
2. 當遍歷到集合的最後一個元素時，我們可以輸出當前的子集，這就是終止條件。

演算法與設計

```cpp
#include <iostream>
using namespace std;

// 遞迴函數來計算集合的冪集
void powerset(char set[], char subset[], int n, int index, int subsetSize) {
    // 基礎情況：當 index 等於集合大小時，輸出當前的子集
    if (index == n) {
        cout << "{";
        for (int i = 0; i < subsetSize; i++) {
            cout << subset[i];
            if (i != subsetSize - 1)
                cout << ",";
        }
        cout << "}" << endl;
        return;
    }

    // 遞迴情況 1：不包含當前元素，直接進入下一層遞迴
    powerset(set, subset, n, index + 1, subsetSize);

    // 遞迴情況 2：包含當前元素，加入子集後進入下一層遞迴
    subset[subsetSize] = set[index];
    powerset(set, subset, n, index + 1, subsetSize + 1);
}

int main() {
    char set[] = { 'a', 'b', 'c' };
    int n = sizeof(set) / sizeof(set[0]);
    char* subset = new char[n];

    cout << "Powerset: " << endl;
    powerset(set, subset, n, 0, 0);
    return 0;
}
```

效能分析

- 空間複雜度：O(n)
- 時間複雜度：O(2^n)

心得

通過這次實作，我對遞迴的應用有了更深入的理解。特別是在處理集合運算時，遞迴的確是一種強大的工具。冪集的生成是一個典型的遞迴問題，它展示了如何使用遞迴來解決二元選擇問題。此外，我也學到了時間與空間複雜度在此類問題中的重要性。生成冪集的時間複雜度為 \(O(2^n)\)，這讓我意識到當集合規模增大時，運算資源的消耗會急劇增加。我相信，如果未來需要優化這類程式，必須考慮如何有效減少計算過程中的冗餘。