Asymptotic Analysis

```
94  void SaveConnections(int k, string node, string parent, map < string, vector < string >> & graph, map < string, int > & Connections) {
95
96      if (k < 0)
97          return;
98
99      if (Connections.find(node) == Connections.end()) {
100         Connections.insert(pair < string, int > (node, k + 1));
101     } else {
102         if (Connections[node] < k + 1) Connections[node] = k + 1;
103     }
104
105     for (const auto & i: graph[node]) {
106
107         if (i != parent) {
108
109             SaveConnections(k - 1, i, node, graph, Connections);
110         }
111     }
112 }
```

برای پیدا کردن کانکشن ها ، ما در این تابع از تابع find که در مپ هست استفاده میکنیم . پیچیدگی این تابع به دلیل Binary Search که در آن اتفاق می افتد برابر $O(Logn)$ می باشد . در ادامه یک حلقه for داریم که در آن تابع بازگشتی صدا زده شده است . به دلیل اینکه تابع بازگشتی درون حلقه for صدا زده شده است و هر تابع بازگشتی خود یک حلقه for دیگر دارد ، تعداد فراخوانی های تابع بازگشتی بسیار مهم است . اینجا به دلیل اینکه k ابتدا برابر 5 است و تابع بازگشتی درون حلقه فور صدا زده شده است ، پیچیدگی این حلقه همانند 5 حلقه for تو در تو میباشد که برابر $O(n^5)$ میشود . پس با توجه به دو پیچیدگی ای که بدست آوردیم ، پیچیدگی کل این تابع برابر $O(n^5)$ می باشد