Análise e Síntese de Algoritmos

Relatório Projecto 1 – *Social Groups*

Autor: João Pinho

Número: 66047

Análise do Problema

O problema proposto, retrata uma sociedade onde existem pessoas e grupos de pessoas que partilham informação entre si. Desta forma, esses grupos, que podem ser de um ou mais elementos, partilham informação de duas maneiras possíveis, ou a partilham de forma isolada somente dentro do grupo ou a partilham para fora do grupo, durante este relatório irei referir-me aos primeiros como grupos não sociais e aos segundos como grupos sociais, isto é, os grupos que socializam com outros são "sociais".

Descrição da Implementação

Para resolver este problema, escolhi a seguinte abordagem: comecei por criar um grafo, construido em função dos inputs fornecidos de exemplo (inicialmente), na criação do grafo, foram criados vértices para cada uma das pessoas, independentemente, de estas partilharem ou receberem partilhas de terceiros, desta forma, se ninguem partilhasse informação, no limite teriamos n grupos correspondentes às n pessoas.

No grafo, as partilhas correspondem aos arcos em cada vértice. Internamente, os vértices do grafo são indexados num array estático criado a partir do input fornecido e os arcos de cada vértice são colocados numa lista ligada existente em cada um destes.

Criado o grafo, o próximo passo envolveu a identificação das componentes fortemente ligadas (SCCs). Sendo o grafo estático e dirigido, e por não existir a necessidade de andar a adicionar e remover arcos/vértices dinamicamente, recorri a um algoritmo eficiente para solucionar esta componente do problema, conhecido como Algoritmo de Tarjan, com uma complexidade assimptótica no pior caso de $O(|V| + |E|)$.

Assim sendo, só com esta informação, o grafo e os SCCs, foi possível obter resposta a algumas das questões pedidas, nomeadamente:

1. *Número de grupos máximos de pessoas que partilham informação;*

   Para extrair esta informação, sem influienciar o tempo de execução do algoritmo de Tarjan, implementei um contador que vai sendo incrementado à medida que as procuras em profundidade do algoritmo vão identificando os SCCs. Desta forma, com muito pouco esforço consegue-se saber precisamente a resposta a esta questão.

2. *Tamanho do maior grupo máximo de pessoas que partilham informação;*

   Durante a execução do algoritmo de Tarjan, existe um passo posterior à identificação de cada SCC que envolve retirar da pilha os elementos contidos no SCC, logo, e mais uma vez, para dar resposta a esta questão em tempo constante (por estar dissimulado no lógica do algoritmo), fui guardando o maior número de elementos de cada SCC, substituindo sempre este número por um superior, quando identificado. Assim, no final da execução do algoritmo, o número máximo de elementos estava disponível automaticamente no contexto da resposta retornado pelo algoritmo.

3. *Número de grupos máximos de pessoas que partilham informação apenas dentro do grupo;*

   Após o cálculo dos SCCs, ficaram guardados numa estrutura auxiliar os vértices de cada um. Desta forma, obter os grupos máximos de pessoas (SCCs) que partilham informação apenas dentro do grupo (SCC), implicou percorrer os arcos de cada vértice de um SCC, para cada SCC, e detectar se esse SCC é ou não "social", ou seja, se possui um arco para um elemento fora do seu próprio SCC. No algoritmo que faz este cálculo, mal um SCC é detectado como sendo "social" a navegação pelos vertices/arcos é abortada e este não é considerado na contagem. No final do algoritmo apenas os SCCs que não tenha a flag "issocial" activa, são considerados.

Complexidade Assimptótica

Em termos de complexidade, vamos analisar uma versão resumida do algoritmo global:

<u>Contrução do Grafo</u>

```
fscanf(stdin, "%d %ld\n", &npeople, &nshares);
socials = graph_new(npeople);

for (
 lineno=0;
 (lineno < nshares) && (fscanf(stdin, "%d %d\n", &sharedfrom, &sharedto) != EOF);
 lineno++
)
    graph_add(socials, sharedfrom, sharedto);
```

Tempo: O(npeople + nshares), onde *npeople* é o número de pessoas envolvidas nas partilhas e *nshares* é igual ao número de partilhas extraidas do input. A adição de arcos é efectuada em tempo constante uma vez que esta operação corresponde a adicionar um arco à *tail* da lista. Uniformizando, temos O(|V| + |E|).

<u>Algoritmo de Tarjan</u>

```
result = tarjan_scc(socials);
```

Tempo: O(|V| + |E|), onde V corresponde ao número de pessoas e E ao número de partilhas.

Classificação do SCCs para determinar os SCCs não sociais

```
int scc_classify_nonsocial(){

      // initializations here

     if(count_scc == 1) return 1;

      for(i=0; i <count_scc; i++){
         issocial=0;

        for(j=scc_s[i]; j<=scc_f[i]; j++){
                the_vertice = vertices[sc_vertices[j]];

                if(the_vertice == NULL || the_vertice->edges == NULL) continue;

                edge = the_vertice->edges->head;
                while(edge != NULL){
                        scc_connect = vertices[(edge->value-1)]->scc_index;

                        if(scc_connect != i){
                           issocial=1;
                            break;
                        }

                        edge = edge->next;
                }

                if(issocial) break;
        }

        if(!issocial) count++;
      }

      // memory release here

      return count;
}
```

Portanto, o código acima é relativamente simples, onde:

- O 1º ciclo *for* corre *m* vezes, onde *m* é o número de grupos máximos de pessoas identificados (SCCs);
    - O 2º ciclo *for* interior corre no pior caso |V| vezes, imagine-se o caso de um SCC com todos os vértices, se bem que isso nunca acontece devido à validação inicial de que, se só existir 1 SCC automaticamente é retornado 1, o que significa que existe 1 grupo que não partilha informação para fora, como é óbvio.
        - O 3º ciclo *while* interior, corre no pior caso |E| vezes, 1 SCC com todos os arcos.

Tempo: O(|V| + |E|), onde no pior caso, temos tantos SCCs, como vértices e nesse caso teriamos O(V^2 + E), mas nesse caso o |V| que vem dos vértices de um SCC nunca seria |V|, mas seria sim, mais próximo de 1, ou seja, a componente |*m*| é majorada por |V|, por isso podemo-mos omiti-la da complexidade assimptótica. E portanto, a complexidade assimptótica fica O(|V| + |E|).

Resumindo, as operações do algoritmo e respectivas complexidades assimptóticas são:

- **Contrução do Grafo** : O(|V| + |E|)
- **Algoritmo de Tarjan** : O(|V| + |E|)
- **Classificação do SCCs para determinar os SCCs não sociais** : O(|V| + |E|)

Tempo Global do Algoritmo: O(3*|V|+|E|) = O(|V|+|E|)

Testes ao Algoritmo

O testes ao algoritmo focaram-se essencialmente em testar dois casos, o caso que que E = V, ou seja, existem tantas partilhas quanto pessoas e o caso E = V^2, onde o número de partilhas é quadrático em relação ao número de pessoas.

Queremos assim demonstrar que para o primeiro caso a complexidade assimptótica vai ser algo do estilo:

> 3 * O(V+E), com E=V temos 3 * O(V+V) = 6 * O(V) = O(V) e portanto iremos obter tempo linear.

Enquanto que no 2º caso, temos uma complexidade assimptótica do estilo:

> 3 * O(V+V^2) = 3 * O(V^2) = O(V^2) e portanto iremos obter tempo quadrático.