TÉCNICO LISBOA

1º Projeto de ASA - 2º Semestre 2013/2014

GRUPO 187 - Carlos Correia nº76512 Gonçalo Mendes nº77044

Breve Introdução

Com o aparecer de cada vez mais vastas redes de informação e de comunicação entre pessoas torna-se necessário encontrar soluções que vão ao encontro das necessidades da comunidade global estabelecendo sempre um compromisso entre eficiência e eficácia dos serviços. Redes sociais como o Facebook que ligam milhões de pessoas necessitam de algoritmos bem definidos, fiáveis, e acima de tudo que não comprometam a estabilidade e a rapidez dos procedimentos para partilhar algo com alguém. Neste tipo de projeto reside uma componente crucial da aprendizagem de qualquer Engenheiro Informático, a capacidade de desenhar ou implementar algoritmos sofisticados para obtenção de resoluções eficientes, ou seja, que cumpram um objectivo num espaço de tempo bem definido.

Solução Implementada

Com base na informação que é cedida relativamente a pessoas e partilhas, o objetivo é a concretização de grupos de partilha e posterior análise para extração de informação relativa a número de grupos, maior grupo, e atomicidade de grupos, ou seja se não partilham informação para o exterior.
A solução implementada visa na **criação de um grafo** dirigido e consequente descoberta de componentes fortemente ligados utilizando o **algoritmo Tarjan**.
A representação por forma de grafo faz com que seja possível identificar grupos de partilha e ligações entre si de uma forma transparente, sendo utilizado o algoritmo sobre o grafo para identificar os SCC's correspondentes a grupos de partilha.

A forma de representar o grafo foi utilizando uma **lista de adjacências.** Esta foi a representação considerada pondo de parte a matriz de adjacências por ocupar menos espaço, ser mais eficiente, e bastar ler a lista de adjacências de um vértice para encontrar os seus vértices adjacentes.

O nosso programa foi codificado em **linguagem C** visto ambos os membros do grupo terem maior experiência com esta, e por apresentar menor grau de abstração relativamente a outras, sendo possível realizar operações com maior liberdade sem restrições comparado a Java, segunda linguagem com qual temos alguma familiaridade.

Posto isto, a nossa lista de adjacências foi construída recorrendo a **duas estruturas e um array**. Existe uma estrutura vertex que guarda apenas um ponteiro para a estrutura edge, estando as suas "instâncias" guardadas no array correspondente ao número de pessoas. A segunda estrutura contem dois campos, o índice do vértice no array e o ponteiro para a mesma estrutura, criando assim uma lista ligada com as adjacências.

Voltando ao algoritmo Tarjan, este foi escolhido por ser o mais trivial e de complexidade linear para encontrar grupos de partilha, os SCC's. A implementação foi baseada no pseudo-código dos slides das teoricas.

Com apenas o algoritmo tarjan clássico e a nossa lista de adjacências apenas foi possível concretizar o **output do número de grupos** e com uma alteração minima o output que devolve o **número do maior grupo**. Utilizando uma variável global "maiorGrupo" e aplicando MAX(maiorGrupo,ultimo_SCC_numero_elementos) dentro do Tarjan_Visit obtemos o output pretendido.

Para adicionar a funcionalidade do teste da atomicidade dos grupos, isto é, **se partilham informação só entre si**, foi concebida uma função que, imediatamente após a descoberta de um SCC por parte do Tarjan_Visit, é chamada com dois argumentos(três para efeitos práticos de recursão), a raiz do SCC e o número do mesmo. Esta função, recursiva, começa na raiz e verifica todas as adjacências dos vértices que derivam desta. Caso chegue de novo à raiz termina normalmente, devolvendo 1. Se encontrar uma adjacência que não pertença ao SCC em causa termina e devolve 0. Para fazer isto, no momento em que os valores da pilha estão a ser libertados é actualizada uma flag de todos os vértices(scc_nr) que contém o nr do SCC em causa. Durante a execução da função recursiva o 2º argumento é comparado esta flag para cada adjacência.

Análise Teórica e Avaliação Experimental

O algoritmo Tarjan clássico corre com complexidade linear de O[V + E] onde V representam os vetrices e E as edges.

O algoritmo criado para testar a atomicidade dos Grupos corre também com complexidade linear de O[V + E] tendo estrutura semelhante ao do Tarjan na medida em que tem uma recursão dentro de uma iteração (a verificação das adjacências de um vértice)

Fazendo diversos testes com número de vértices e arcos a variar, foi possível provar que o algoritmo corre de facto a tempo linear, dado que para um input n(V vertices+ E arcos) o tempo de execução é t, e para um input de 2n vértices o tempo de execução é aproximadamente 2t.