

## Problem Set no. 2 — Global minimum cuts

Given: November 25, 2009

**Exercise 2.1** Let  $G = (V, E)$  be a weighted undirected graph with  $|V| \geq 3$ . Show that there are at least two distinct vertices  $t_1, t_2 \in V$  such that  $(V - \{t_1\}, \{t_1\})$  is a minimum  $s_1$ - $t_1$  cut, for some  $s_1 \in V$ , and  $(V - \{t_2\}, \{t_2\})$  is a minimum  $s_2$ - $t_2$  cut, for some  $s_2 \in V$ . Find graphs in which there are no three such vertices.

**Exercise 2.2** Does the algorithm of Stoer and Wagner for finding a minimum  $s$ - $t$  cut, for some  $s, t \in V$ , work also for *directed* graphs? If not, what goes wrong?

**Exercise 2.3** Find a graph on  $n$  vertices that has  $\binom{n}{2}$  global minimum cuts.

**Exercise 2.4** Show that one run of the simple random contraction algorithm of Karger and Stein on a weighted graph can be implemented in  $O(n^2)$  time. (Hint: Maintain the sum of the edge weights incident on each vertex.)

**Exercise 2.5** Show that one run of the simple random contraction algorithm of Karger and Stein on an initially unweighted graph can be implemented in  $O(m)$  time. (Hint: Start by choosing a random permutation of the edges. Next, contract the first  $m/2$  edges and check whether there are at least two vertices in the contracted graph.)

**Exercise 2.6** Consider the following algorithm for finding a minimum global cut in an unweighted undirected graph on  $n$  vertices: Choose a random edge and contract it. Repeat this operation until the number of remaining vertices in the graph is  $t$ , where  $t$  is a parameter to be determined later. Then, apply a deterministic algorithm for finding a global minimum cut in the resulting graph. The complexity of the deterministic minimum cut algorithm is  $O(n^a)$ , where  $n$  is the number vertices in the graph. Finally, run this combined algorithm a sufficient number of times so that the probability that it finds a global minimum cut is at least  $1/2$ . What is the optimal choice of  $t$  and what is then the running time of this algorithm? What happens if you use this algorithm instead of the  $O(n^a)$  algorithm?

**Exercise 2.7** A cut in a graph  $G = (V, E)$  is said to be  $\alpha$ -minimal, for  $\alpha \geq 1$ , if its size is at most  $\alpha$  times the size of a global minimum cut. Obtain an upper bound on the number  $\alpha$ -minimal cuts that a graph on  $n$  vertices can have.

**Exercise 2.8** Show that a simple variant of the random contraction algorithm of Karger and Stein can be used to find a *minimum 3-cut* of an undirected and unweighted graph  $G = (V, E)$  on  $n$  vertices. (A 3-cut of  $G = (V, E)$  is a partition of  $V$  into three non-empty sets  $A, B$  and  $C$ . The size of the cut is the number of edges connecting vertices from different sets.) What is the success probability of the algorithm? Extend your answers to the case of minimum  $k$ -cuts.

**Exercise 2.9** One of the students of the course suggests the following algorithm for finding a minimum  $s$ - $t$  cut in an undirected and unweighted graph  $G = (V, E)$  on  $n$  vertices: Choose, uniformly at random, an edge that does *not* connect  $s$  and  $t$ , and contract it. If the contracted edge touches  $s$ , then the newly formed vertex continues to play the role of  $s$ . The same goes for  $t$ . This goes on until only two vertices are left in the graph. These must be  $s$  and  $t$ , and an  $s$ - $t$  cut is obtained. Find graphs in which the success probability of this algorithm is exponentially small.