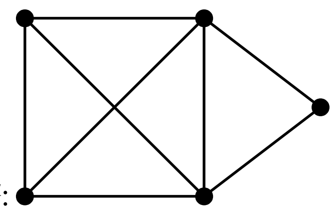
Martin Ziegler
Sewon Park, Junhee Cho, Ivan Koswara

Issued on May 25, 2018
Solutions due: June 6, 2018

CS500

Spring 2018, Assignment #6

PROBLEM 14 (1+2+2+2+3P) **:**
Recall that *Vertex Cover* is the following minimization problem: Given an undirected graph $G = (V,E)$, find the least number $k = k(G)$ of vertices $v_1,\dots,v_k \in V$ such that every edge $e \in E$ is incident to (i.e. has among its two end points) at least one vertex from the set $C = \{v_1,\dots,v_k\}$.



a) Determine $k(G)$ and an optimal Vertex Cover for the following graph G:

b) The lecture had established the following greedy algorithm, initialized with $C := \{\} =: F$, to yield a 2-approximation to Vertex Cover:

> For each edge $e = \{a,b\} \in E$, put e into F and put *both* a,b into C
> and remove from E all edges incident to a or b.

Prove that analysis tight by constructing (a family of) graphs G where the algorithm produces a vertex cover of size $\geq 2k(G)$.

c) What about a modified heuristic putting only *one* arbitrary of each edge's end points into C?

d) *Bin Packing* is the following optimization problem: Given a "maximal weight" $W \in \mathbb{N}$ and m integer "packets' weights" $w_1,\dots,w_m \leq W$, distribute these packets into as few "bags" as possible such that each bag gets to carry weight not exceeding W. Let $B = B(W;w_1,\dots,w_m) \in \mathbb{N}$ denote this least number of bags.
Determine $B(10;5,4,3,2,2,5,4,3,5,4,3)$.
Run the *First Fit* algorithm from Item (e) on this example.

e) Prove that the following greedy *First Fit* algorithm yields a factor 2 approximation, i.e. uses at most $2B$ bags:

> For each $j = 1,\dots,m$ place packet $\#j$ into the first bag it fits into;
> otherwise open a new bag.

PROBLEM 15 (2+2+2+2+2P) **:**
Recall from the lecture the definition of the (0/1) Knapsack maximization problem and its relaxation by approximation.

a) The lecture introduced an algorithm computing the *value* of an optimal packing by means of *dynamic programming*.
Modify it such as to *determine* an optimal packing, that is, a 0/1 vector indicating which packets to include and which not.

b) Implement in ELICE (`http://kaist.ELICE.io`) the algorithm from a).

c) Determine the optimal solutions to the following instances:

   i) values=$(1,3,2,1,4)$, weights=$(3,4,3,3,6)$, knapsack capacity 11.

   ii) values=$(135,139,149,150,156,163,173,184,192,201,210,214,221,229,240)$, weights=$(70,73,77,80,82,87,90,94,98,106,110,113,115,118,120)$, capacity=750.

   iii) values=$(825594,1677009,1676628,1523970,943972,97426,69666,1296457, 1679693,1902996,1844992,1049289,1252836,1319836,953277,2067538, 675367,853655,1826027,65731,901489,577243,466257,369261)$, weights=$(382745,799601,909247,729069,467902,44328,34610,698150, 823460,903959,853665,551830,610856,670702,488960,951111, 323046,446298,931161,31385,496951,264724,224916,169684)$, capacity=6404180.

d) Building on (b), implement in ELICE the approximation *scheme* from the lecture via rounding.

e) Determine an approximate solution up to error $\varepsilon = 0.001$ to the instance from c iii).