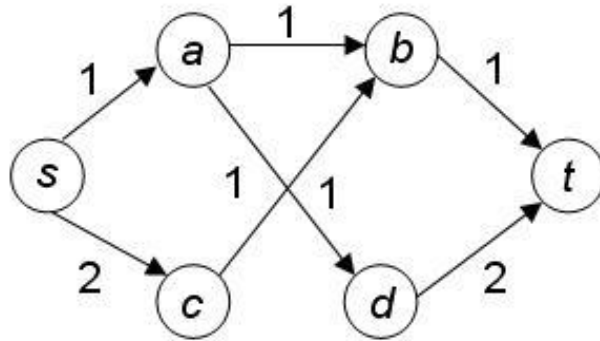


### Problem A (15%)

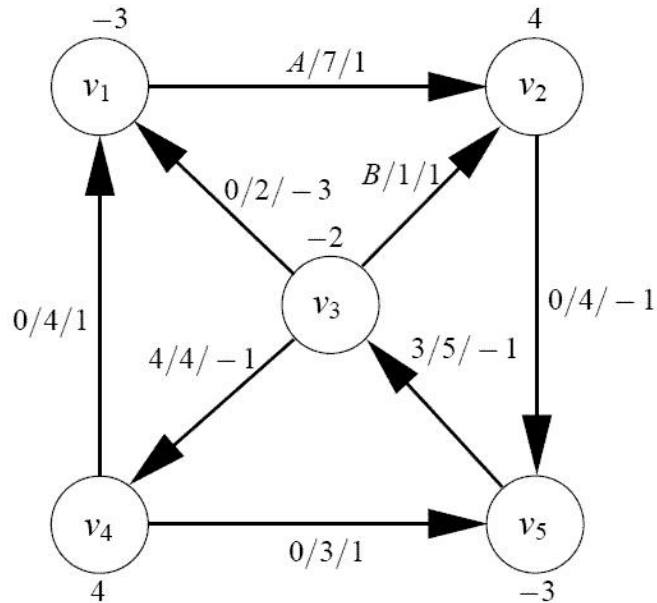
In the flow network illustrated below, each arc is labeled with its capacity. We are using the Ford-Fulkerson method to find a maximum flow. The first augmenting path we choose is  $s$ - $a$ - $b$ - $t$ .



1. Draw the residual network resulting by updating the flow using this augmenting path.
2. List all the augmenting paths that could be used for the second augmentation step.
3. What is the value of a maximum flow in the network?
4. State, as a pair of sets, a minimum cut in the network.

### Problem B (15%)

In the following picture a flow network  $G$  with costs *and* a feasible flow  $f$  in  $G$  is given.



Each node is labeled with its balance. The three numbers stated along each arc are (in order) the *flow* along this arc, the *capacity* of this arc and the *cost* of this arc. For instance,  $b_{v_1} = -3$ ,  $f_{(v_5, v_3)} = 3$ ,  $c_{(v_5, v_3)} = 5$  and  $k_{(v_5, v_3)} = -1$ . Flows along the two arcs  $(v_1, v_2)$  and  $(v_3, v_2)$  are not stated and have been replaced with the symbols  $A$  and  $B$  instead. Other flows and costs that are not stated are as given by the skew symmetry rules, or they are zero.

1. For which values of  $A$  and  $B$  is the stated flow a feasible flow?
2. Draw the residual network  $G_f$  (with  $A$  and  $B$  being those values found in question 1).
3. List all augmenting cycles in  $G_f$  (indicate each cycle as a list of vertices).

### Problem C (20 %)

Consider the following five linear programming dictionaries:

Dictionary a:

$$\begin{array}{r} x_4 = 4 - 2x_1 + 3x_2 - x_3 \\ x_5 = 3 - 3x_1 + x_2 - 2x_3 \\ x_6 = 3 - 3x_1 + 2x_2 - 2x_3 \\ \hline z = 1 + x_1 + 2x_2 + 4x_3 \end{array}$$

Dictionary b:

$$\begin{array}{r} x_2 = 1 - 2x_1 - 3x_3 - x_4 \\ x_5 = -1 - 4x_1 - x_3 - 2x_4 \\ x_6 = 2 - 3x_1 - 4x_3 - 2x_4 \\ \hline z = 3 + 2x_1 - 4x_3 - 2x_4 \end{array}$$

Dictionary c:

$$\begin{array}{r} x_2 = 2x_1 - 3x_3 - x_4 \\ x_5 = -x_1 - 2x_3 - 2x_4 \\ x_6 = 8 - 3x_1 - 4x_3 - 2x_4 \\ \hline z = 1 - 2x_1 - 4x_3 - 8x_4 \end{array}$$

Dictionary d:

$$\begin{array}{r} x_1 = -2x_3 + 10x_4 - x_6 \\ x_2 = 2 - 4x_3 - 10x_4 - 2x_6 \\ x_5 = -3x_3 - x_4 - 2x_6 \\ \hline z = 7 - x_3 + x_4 + 2x_6 \end{array}$$

Dictionary e:

$$\begin{array}{r} x_1 = 2 - 10x_3 + 10x_4 - x_6 \\ x_2 = -x_3 - 10x_4 - 2x_6 \\ x_5 = -10x_3 - x_4 - 2x_6 \\ \hline z = 7 + x_3 + x_4 - 2x_6 \end{array}$$

Copy the table on the following page and fill it in as indicated below:

- In the row labeled “Feasible”, indicate for each dictionary if it is feasible. Write **Y** if it is, and write **N** if it is not.
- In the row labeled “Terminal”, indicate for each dictionary if it is terminal, i.e., if it is a final dictionary in the execution of the simplex algorithm *and* the corresponding basic solution is optimal. Write **Y** if it is, and write **N** if it is not. Write **N/A** if the present dictionary is not feasible.
- In the row labeled “Degenerate”, indicate for each dictionary if the corresponding basic solution is degenerate. Write **Y** if it is, and write **N** if it is not. Write **N/A** if the present dictionary is not feasible.

- In the row labeled “Entering”, write for each dictionary the *name* of the variable who will enter the basis in the next iteration of the simplex algorithm, *assuming that Bland’s rule is applied*. Write **N/A** if there is no such variable or if the present dictionary is not feasible.
- In the row labeled “Leaving”, write for each dictionary the *name* of the variable that will *leave* the basis in the next iteration of the simplex algorithm, *assuming that Bland’s rule is applied*. Write **N/A** if there is no such variable or if the present dictionary is not feasible.
- In the row labeled “New value”, write the value of the objective function for the basic solution corresponding to the dictionary obtained *after* performing one pivot according to Bland’s rule. Write **N/A** if there is no such next dictionary or if the present dictionary is not feasible.

	a	b	c	d	e
Feasible					
Terminal					
Degenerate					
Entering					
Leaving					
New value					

## Problem D (15 %)

Let  $U$  be a finite set. Given a finite family  $S_1, S_2, \dots, S_k$  of finite subsets of  $U$ , a subset  $H$  of  $U$  is called a *hitting set* of the family, if and only if it has non-empty intersection with all of the sets in the family, i.e.,

$$\forall i : S_i \cap H \neq \emptyset$$

As an example, let  $U = \{1, 2, \dots, 10\}$  and let the family be  $S_1 = \{1, 2, 5\}$ ,  $S_2 = \{1, 2, 7\}$ ,  $S_3 = \{2, 7, 8, 10\}$ . Then  $H = \{1, 7\}$  is a hitting set for the family. Indeed,  $S_1 \cap H = \{1\}$ ,  $S_2 \cap H = \{1, 7\}$  and  $S_3 \cap H = \{7\}$ , so all three intersections are non-empty.

In this problem, we are interested in finding the *smallest* possible hitting set for a given family. For instance, the smallest hitting set for the example is the singleton  $\{2\}$ .

1. Given a family, show how to express as an integer linear program the problem of finding a smallest possible hitting set for the family. Explicitly state what your program would look like for the stated example and what the optimum solution (as an integer vector) for this example would be.

## Problem E (15 %)

Dr. McDonald has measured the weights and heights of 100 students. The collected data are  $w_i$  (the weight of student no.  $i$  in kilograms) and  $h_i$  (the height of student no.  $i$  in centimeters), for  $i = 1..100$ . Dr. McDonald wishes to find constants  $a$  and  $b$ , so that the weight  $w_i$  of any student can be predicted from the height  $h_i$  of that student using the formula  $ah_i + b$ . Unfortunately, plotting his data, he finds that it is not possible to find constants which would yield an exact prediction in all 100 cases. He now considers two possibilities.

- He could try to find constants  $a$  and  $b$  minimizing the *worst case* error of the prediction, i.e., the number

$$e = \max_{i=1}^{100} |w_i - (ah_i + b)|,$$

where  $|x|$  is the absolute value of  $x$ . That is,  $|x| = x$  when  $x \geq 0$  and  $|x| = -x$  when  $x < 0$ . Show how to find such constants  $a, b$  using linear programming.

- He could try to find constants  $a$  and  $b$  minimizing the *average case* error of the prediction. i.e., the number

$$e' = \frac{1}{100} \sum_{i=1}^{100} |w_i - (ah_i + b)|.$$

Show how to find such constants  $a, b$  using linear programming.

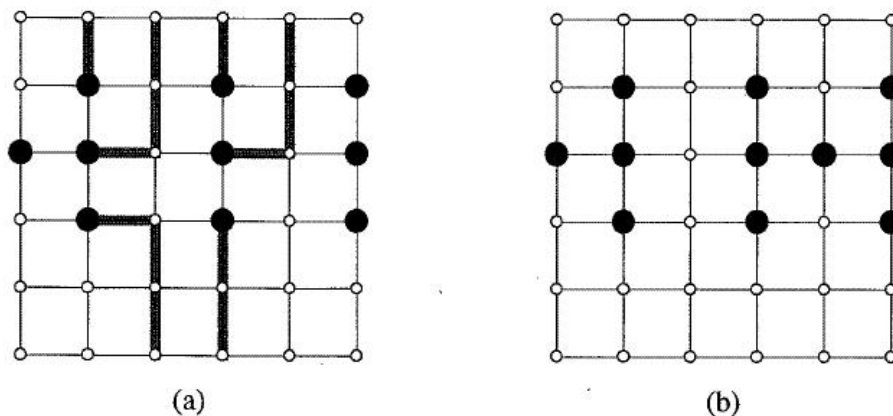
## Problem F (20 %)

Note that this problem is similar to Problem 26-1 of Cormen, Leiserson, Rivest and Stein, and that the figure has been borrowed from there, but that this problem is not identical to that problem.

An  $n \times n$  grid is an undirected graph consisting of  $n$  rows and  $n$  columns of vertices, as shown in Figure 26.11. We denote the vertex in the  $i$ th row and the  $j$ th column by  $(i, j)$ . All vertices in a grid have exactly four neighbours, except for the boundary vertices, which are the points  $(i, j)$  for which  $i = 1$ ,  $i = n$ ,  $j = 1$ , or  $j = n$ . Given a set  $S$  of  $m \leq n^2$  starting points in the grid, the *cheapest escape problem* is to

1. determine whether or not there are  $m$  vertex-disjoint paths from the starting points of  $S$  to any  $m$  different points on the boundary *and*
2. if there is such an *escape*, to find the one where the paths use the fewest possible *total* number of edges. This number we call the *cost* of the escape.

For example, in the following picture, two instances of the cheapest escape problem are given. The starting points are the black vertices. For the instance (a), an escape is indicated. It has cost  $3+3+3+2+1+1+0+0+0+0=13$ . The instance (b) does not have any escape.



1. Show how to model the cheapest escape problem as a min cost flow problem in such a way that the problem can be solved efficiently by using a standard min cost flow algorithm.