Exercise for Analysis of Algorithms

Exercise 5

If a flow diagram consists of n nodes and m edges, how many fundamental cycles do we get?

Solution:

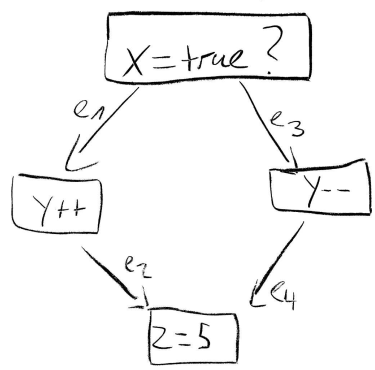
Any spanning tree of a graph on n nodes has to use exactly $n-1$ edges. that means that there are $m-(n-1)$ edges that are not part of the spanning tree. Since every edge not part of the spanning tree is part of exactly one fundamental cycle we get $m-n+1$ many fundamental cycles.

Exercise 6

Prove or disprove: In every flow diagram you can find a spanning tree such that all fundamental cycles contain only edges that are labeled with plus.

Solution:

Consider a part of a program that contains an if-else statement.



A spanning tree of that structure always has one edge not part of the tree, no matter what edge is selected, we always have to use the two edges of the other side in the opposite direction. So without loss of generality let e_1 be the non tree edge, e_2 the edge on the same side (either before or after e_1) and e_3 and e_4 the two edges on the other side. We get

$$C_1 = e_1 + e_2 - e_3 - e_4$$

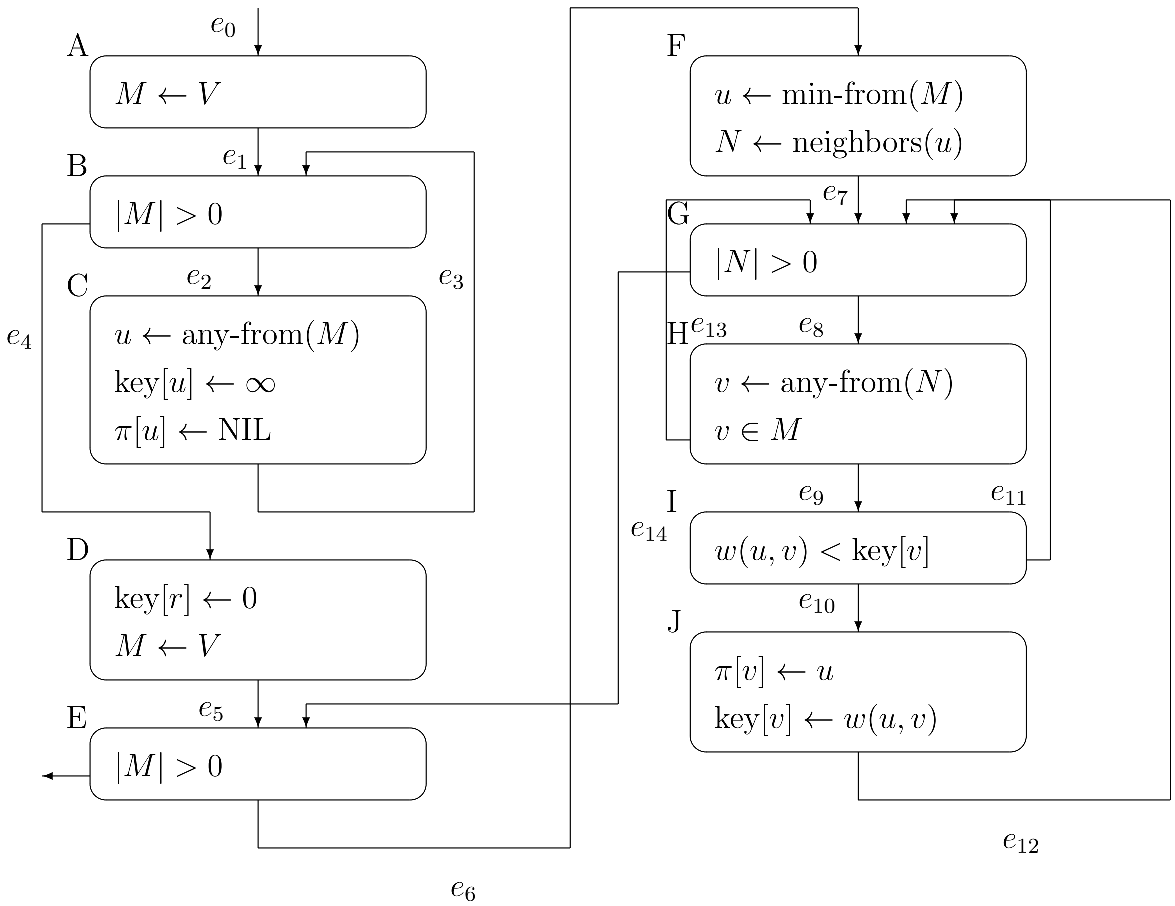
Which thereby disproves the conjecture.

Exercise 7
In this exercise, we consider Prim's Algorithm, which computes a minimum spanning tree. The input to this algorithm is a graph $G = (V, E)$, a weight function on the edges $w : E \to \mathbf{R}$ and a starting node r.

```
for each u ∈ V do
    key[u] ← ∞
    π[u] ← NIL
key[r] ← 0
M ← V
while (M ≠ ∅) do
    u ← min-from(M)
    for each v ∈ neighbors(u) do
        if (v ∈ M) ∧ (w(u, v) < key[v]) then
            π[v] ← u
            key[v] ← w(u, v)
```

Construct the control flow graph, a spanning tree in the control flow graph, the fundamental cycles, a corresponding linear system of equations and a solution to this system.

Solution:
The flow diagram is depicted below. The **for**-loops were changed, since the initializing and iteration condition must be separated.



We choose the spanning tree $e_1, e_2, e_4, e_5, e_6, e_7, e_8, e_9, e_{10}$. This yields the following fundamental cycles:

$$
\begin{aligned}
C_0 &= e_0 + e_1 + e_4 + e_5 \\
C_3 &= e_3 + e_2 \\
C_{11} &= e_{11} + e_8 + e_9 \\
C_{12} &= e_{12} + e_8 + e_9 + e_{10} \\
C_{13} &= e_{13} + e_8 \\
C_{14} &= e_{14} + e_6 + e_7
\end{aligned}
$$

We now use standard linear algebra to find a good set of blocks whose number of visits we need to compute: By E_i, $0 \le i \le 14$, we denote the number of times the program flow visits the edge e_i. With each fundamental cycle above we identify a vector C_i. Then the E_i can be written as a linar combination of the fundamental cycles, i.e.,

$$
(C_0, C_3, C_{11}, C_{12}, C_{13}, C_{14}) \begin{pmatrix} \lambda_1 \\ \lambda_2 \\ \lambda_3 \\ \lambda_4 \\ \lambda_5 \\ \lambda_6 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 1 & 1 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} \lambda_1 \\ \lambda_2 \\ \lambda_3 \\ \lambda_4 \\ \lambda_5 \\ \lambda_6 \end{pmatrix} = \begin{pmatrix} E_0 \\ E_1 \\ E_2 \\ E_3 \\ E_4 \\ E_5 \\ E_6 \\ E_7 \\ E_8 \\ E_9 \\ E_{10} \\ E_{11} \\ E_{12} \\ E_{13} \\ E_{13} \end{pmatrix}
$$

for appropriate values of $\lambda_1, \ldots, \lambda_6$. We select six independent rows and obtain the equation

$$
\begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} \lambda_1 \\ \lambda_2 \\ \lambda_3 \\ \lambda_4 \\ \lambda_5 \\ \lambda_6 \end{pmatrix} = \begin{pmatrix} E_0 \\ E_2 \\ E_{11} \\ E_{12} \\ E_{13} \\ E_{14} \end{pmatrix} = \begin{pmatrix} 1 \\ C \\ I - J \\ J \\ H - I \\ G - H \end{pmatrix}.
$$

This means, we only need to compute the values of C, G, H, I, J for a complete analysis ($E_0 = 1$ is trivally known), and then all other values can be derived: We see that $E_0 = E_1 = E_4 = E_5$, $E_2 = E_3$, $E_6 = E_7 = E_{14}$, $E_{10} = E_{12}$. This implies $A = 1$, $B = E_1 + E_3 = C + 1$, $D = E_4 = 1$, $E = E_5 + E_{14} = 1 + G - H$, $F = E_6 = G - H$.

Exercise 8

Let $w \in \{a, b\}^n$ a word that has been chosen uniformly at random. How often is the body of the `while`-loop executed on average in the following algorithm? The function `is_palindrome` tests whether a word is a palindrome, i.e., the same when read backwards.

```
i = 2;
while (i <= n)
  if (is_palindrome(w[1],...,w[i]))
      return true;
  i++;
return false;
```

Solution:
Without loss of generality let $w[1]$ be a. If $w[2]$ also is a, which happens with probability $\frac{1}{2}$, we have found a palindrome after one round, and are done, otherwise, we have the prefix ab. In the following round we find a palindrome if an a occurs, otherwise, we have the prefix abb. It follows that we find a palindrome at that moment when a second a occurs. For $1 \leq k \leq n-2$, body is executed k times if and only if $w[2], \ldots, w[k] = b$ and $w[k+1] = a$. The body is executed $n-1$ times if and only if $w[2], \ldots, w[n-1] = b$, as the algorithm terminates after the last round no matter if it found a palindrome or not. The expected number of iterations therefore is

$$E = \Big(\sum_{k=1}^{n-2} \frac{1}{2^k} k\Big) + \frac{1}{2^{n-2}}(n-1).$$

We use the fact that

$$\sum_{i=1}^{b} \frac{1}{2^i} = 1 - \frac{1}{2^b}$$

to rewrite the first part of the sum

$$\begin{aligned}
\sum_{k=1}^{n-2} \frac{1}{2^k} k &= \sum_{k=1}^{n-2} \sum_{l=1}^{k} \frac{1}{2^k} \\
&= \sum_{k=1}^{\infty} \sum_{l=1}^{\infty} [1 \leq k \leq n-2][1 \leq l \leq k] \frac{1}{2^k} \\
&= \sum_{k=1}^{\infty} \sum_{l=1}^{\infty} [1 \leq l \leq k \leq n-2] \frac{1}{2^k} \\
&= \sum_{l=1}^{n-2} \sum_{k=l}^{n-2} \frac{1}{2^k} \\
&= \sum_{l=1}^{n-2} \frac{1}{2^{l-1}} \sum_{k=1}^{n-l-1} \frac{1}{2^k} \\
&= \sum_{l=1}^{n-2} \frac{1}{2^{l-1}} (1 - \frac{1}{2^{n-l-1}}) \\
&= \frac{n-2}{2^{n-2}} + \sum_{l=1}^{n-2} \frac{1}{2^{l-1}} \\
&= \frac{n-2}{2^{n-2}} + 2 - \frac{1}{2^{n-3}}
\end{aligned}$$

We put both summands toghether to get the final result

$$E = \frac{n-2}{2^{n-2}} + 2 - \frac{1}{2^{n-3}} + \frac{n-1}{2^{n-2}} = 2 - \frac{1}{2^{n-3}} + \frac{1}{2^{n-2}} = 2 - \frac{1}{2^{n-2}}.$$

Exercise 9

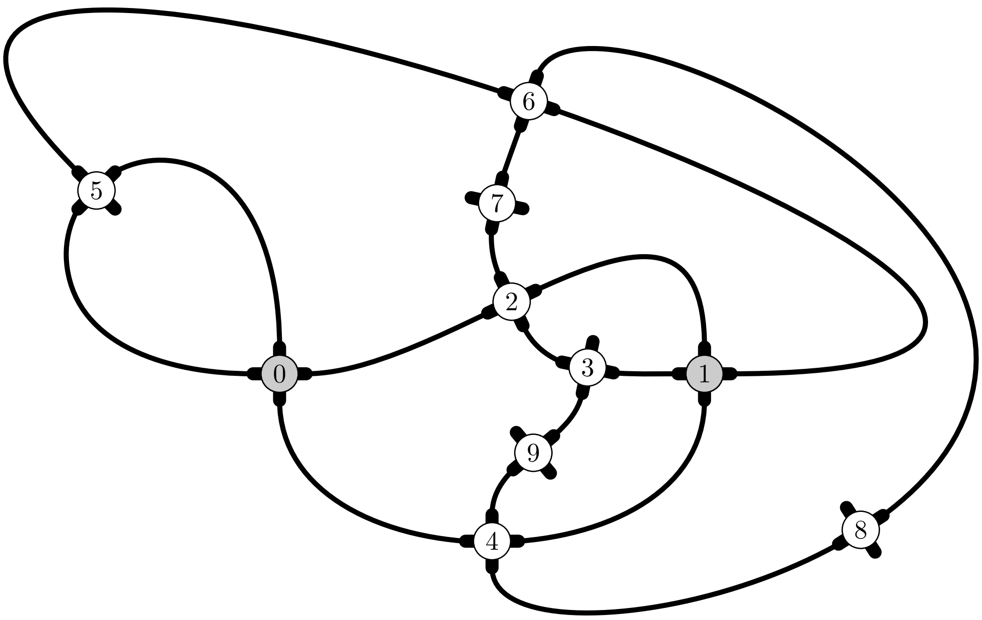
RWTH Aachen University has an exclusive contract with the well-known *Uranus Corporation* on the delivery of canal and pump supplies. Unfortunately, the responsible person failed to notice that Uranus sells only the “one-fits-all” product KKuRPSE (Kanalkopplungsundregulierungspumpstationeinheit). Such a KKuRPSE is a cylinder with four connectors placed north, east, south, and west of the cylinder (see the figure).

Now, the Institute for Tunnel and Canal Construction (ITCC) is conducting some research: The researchers first place n KKuRPSEs on a big green. Then they start to connect KKuRPSEs as follows: They digg a new canal between two arbitrary connectors. The new canal cannot cross another existing canal, of course. They then place a new KKuRPSE somewhere into this new canal, such that exactly two connectors on opposite sides of the new KKuRPSE are now attached to the canal.

The question is: How often can this connection procedure be repeated depending on the number n of initial KKuRPSEs on the green?

Example

Assume the researchers start with two KKuRPSEs, here labeled 0 and 1. Then a couple of connection steps are executed, where each new KKuRPSE receives consecutive numbers until no more connections are possible.



Solution:

First notice the following invariants:

- The number of free connectors at each time is $4n$, since each KKuRPSE uses two connectors, but at the same time introduces two new connectors.

- In each face of the underlying planar graph there is at least one free connector.

Let f be the number of faces, v be the number of vertices, e be the number of edges, and c be the number of connected components at an arbitrary round of the game. Since the graph is planar,

$$v + f = 1 + e + c$$

by a theorem of Euler, and since in each round two new edges are introduced,

$$v = n + \frac{e}{2}.$$

The game ends when in each face only one connector remains, i.e., when $f = 4n$. At this time, the graph is connected, i.e., $c = 1$, due to the outer face. Since in each round exactly two edges are introduced,

$$n + \frac{e}{2} + 4n = 1 + e + 1$$

yields

$$\frac{e}{2} = 5n - 2,$$

and each game lasts exactly $5n - 2$ rounds.