Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

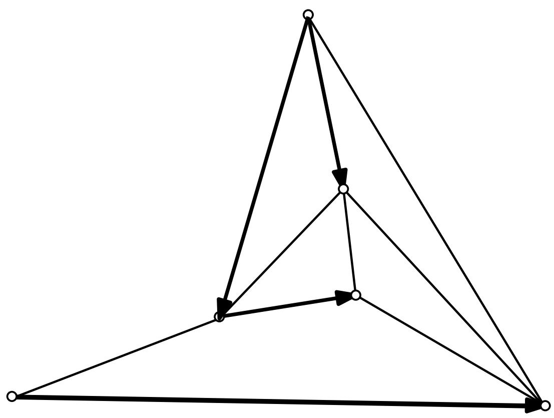
Institute of Theoretical Computer Science
Mohsen Ghaffari, Angelika Steger, David Steurer, Emo Welzl, Peter Widmayer

Algorithms, Probability, and Computing **Exercises KW49** **HS18**

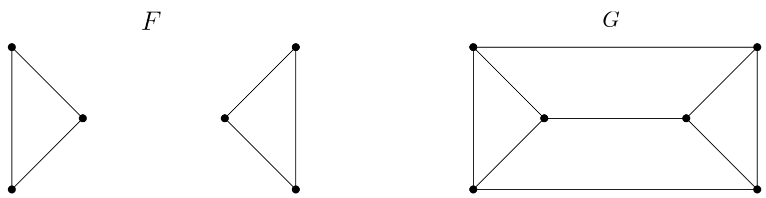
The following exercises will be discussed in the exercise class on December 5, 2018. These are “in-class” exercises, which means that we do not expect you to solve them before the exercise session. Instead, your teaching assistant will solve them with you in class.

Exercise 1

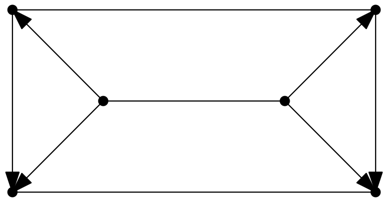
(a) Complete the following partial orientation (four edges are already oriented) to a Pfaffian orientation.



(b) Recall that every planar graph G has a Pfaffian orientation. Consider the graphs $F = (V_F, E_F)$, $G = (V_G, E_G)$, with $V_F = V_G$, $E_F \subseteq E_G$ as given below. Show that not every Pfaffian orientation $\vec{F} = (V_F, E_{\vec{F}})$ is extendable to a Pfaffian orientation $\vec{G} = (V_G, E_{\vec{G}})$, i.e., a Pfaffian orientation $\vec{G}$ of G such that the restriction of $\vec{G}$ to E_F is $\vec{F}$.



Hint: Use the orientations picture below, but do not forget to argue why you can do this.



(c) Show that a graph containing a complete bipartite $K_{2,3}$ cannot be oriented in such a way that every even cycle is oddly oriented. (Note that this includes planar graphs.)

(d) Show that $K_{3,3}$ cannot be oriented in such a way that all nice cycles are oddly oriented.

(e) Show that $\vec{G}$ is Pfaffian iff every nice cycle in G is oddly oriented in $\vec{G}$. (That is, prove the other direction of Lemma 5.5.)

Exercise 2

Show that the maximum of n entries can be computed in $O(\log\log n)$ time-steps, using the CRCW version of PRAM with n processors.

Exercise 3

Use Brent's principle to determine the smallest number of processors that would allow us to the Parallel Prefix algorithm which we saw in the lecture in $O(\log n)$ time. Recall that algorithm had $O(\log n)$ depth and $O(n)$ total computation. Explain how the algorithm with this small number of processors works (i.e., what each processor needs to do exactly).