Exercise sheet 12.

Advanced Algorithms WiSe 2020/21

Instructors: László Kozma, Mahmoud Elashmawi

Due 18:00, February 14th, 2021

Exercise 1 Paths and cycles 4×2 *pts*

Consider the following five problems. Assuming that problem (a) is NP-hard, show that the other four are also NP-hard, by finding suitable polynomial-time reductions.

(a) Directed Hamiltonian Cycle
Input: a directed graph G.
Output: *yes* if G has a Hamiltonian cycle, *no* otherwise.

(b) Directed Long Path
Input: a directed graph G, vertices s and t, integer k.
Output: *yes* if G has a simple path from s to t of length at least k, *no* otherwise.

(c) Undirected Hamiltonian Cycle
Input: an un directed graph G.
Output: *yes* if G has a Hamiltonian cycle (a simple cycle visiting all vertices), *no* otherwise.

(d) Traveling salesperson
Input: n cities and pairwise distances d_{ij} (arbitrary non-negative integers), value N.
Output: *yes* if there is a tour visiting each city exactly once, with total distance at most N, *no* otherwise.

(e) Shortest simple path
Input: graph G with edge weights (possibly negative; negative cycles also allowed), vertices s and t, and a value N.
Output: *yes* if there is an s-t path of length at most N, visiting each vertex *at most once*, no otherwise.

Exercise 2 Variations

(a) (3 pts) The kSAT problems ask, given a boolean CNF formula Φ with each clause of size exactly k, whether there is a variable-assignment so that Φ evaluates to true.

Assuming that 3SAT is NP-complete, argue that 4SAT, 5SAT, etc. are also NP-complete.

(b) (3 pts) The kCOLORING problems ask, given an undirected graph G, whether we can assign each vertex a unique color from a set of k colors, so that no two adjacent vertices have the same color.

Assuming that 3COLORING is NP-complete, argue that 4COLORING, 5COLORING, etc. are also NP-complete.

(c) (3 pts) We have 2COLORING $\in P$, and 2SAT $\in P$. Show one of these claims.

Exercise 3 Vertex cover

Consider the following problem:

VERTEX COVER

Input: an undirected directed graph $G = (V, E)$, a natural number k.
Output: *yes* if there is a subset $V' \subseteq V$ of size at most k, such that every edge in E has at least one endpoint in V'.

(a) (2 pts) Argue that if $T \subseteq V$ is an *independent set*, then $V \setminus T$ hits all edges.

(b) (3 pts) Using this observation, show that VERTEX COVER is NP-complete.

(c) (Extra 3 pts) Show that VERTEX COVER in bipartite graphs is solvable in polynomial-time.

(d) (Extra 4 pts) Show that VERTEX COVER remains NP-complete in planar graphs.

Total: 22 points. Have fun with the solutions!

From: Yumeng Li and Thore Brehmer

Exercise 1 Paths and cycles 4×2 *pts*

Consider the following five problems. Assuming that problem (a) is NP-hard, show that the other four are also NP-hard, by finding suitable polynomial-time reductions.

(a) Directed Hamiltonian Cycle
Input: a directed graph G.
Output: *yes* if G has a Hamiltonian cycle, *no* otherwise.

side Reduction

$$\underset{L_1}{\text{Ham cycle}} \leq_p \underset{L_2}{\text{Ham path}}$$

Input Graph $G(E,V)$

f: pick one vertice $s \in V$

- delete the edge from s to $t \in V$
- ~~(s,t) input vertices~~ Ham. path does not take input vertices, only the graph

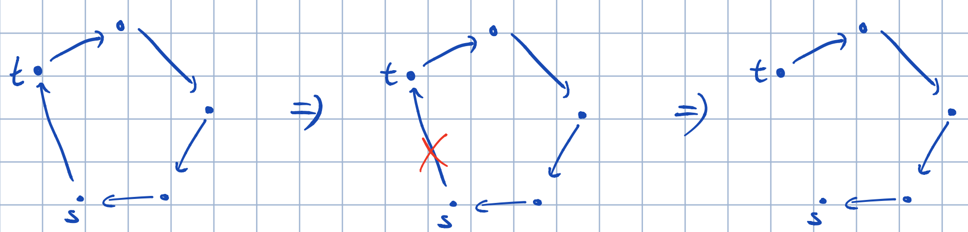
$w \in L_1 \overset{?}{\Rightarrow} f(w) \in L_2$

works, because a Ham cycle has a $|V|$ long simple path if we remove one random edge incorrect

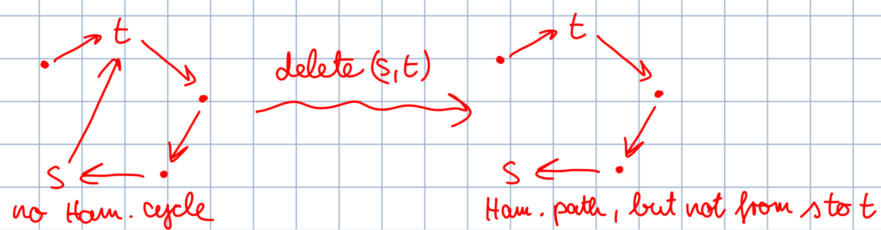
$w \in L_1 \Leftarrow f(w) \in L_2$

works, because if we have a Ham path from s to t, we have a Ham cycle with the edge (s,t)

e.g. is there a Ham Cycle ? $\Leftrightarrow$ is there a Ham path?



Counterexample :



(b) Directed Long Path
Input: a directed graph G, vertices s and t, integer k.
Output: *yes* if G has a simple path from s to t of length at least k, *no* otherwise.

$$\underset{L_1}{\text{Ham Path}} \le_p \underset{L_2}{\text{Long Path}}$$

Input Graph $G(E, V)$ ~~, s, t~~ s and t to be specified in f

f:

- change k to $|V|$

$w \in L_1 \overset{?}{\Rightarrow} f(w) \in L_2$

works, because a Ham Path ~~has~~ is a $|V|$ long simple path

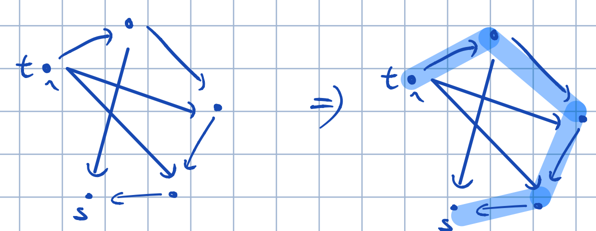
$w \in L_1 \Leftarrow f(w) \in L_2$

works, because if we have a $|V|$ long simple path from s to t, we have a Ham

ff

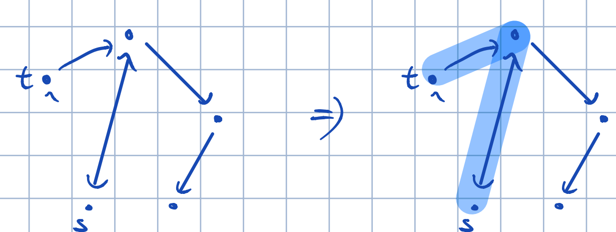
e.g. is there a Ham Path? $\Leftrightarrow$ is there a $k(|V|)$ long path?

1. yes-instant



$|V|$ long Path
$\Rightarrow$ yes

2. No-instant



2 long simple Path

$2 \le |V|$

$\Rightarrow$ no

1

(c) Undirected Hamiltonian Cycle
Input: an un directed graph G.
Output: *yes* if G has a Hamiltonian cycle (a simple cycle visiting all vertices), *no* otherwise.

L_1 $\qquad\qquad$ L_2

Ham Cycle $\leq_p$ undirected Ham Cycle

Input Graph $G(E, V)$

f: for every vertice which has one incoming and one outgoing edge:

- change the outgoing edge to an undirected edge

what about the remaining edges?

all edges in the resulting graph have to be undirected

$w \in L_1 \overset{?}{\Rightarrow} f(w) \in L_2$

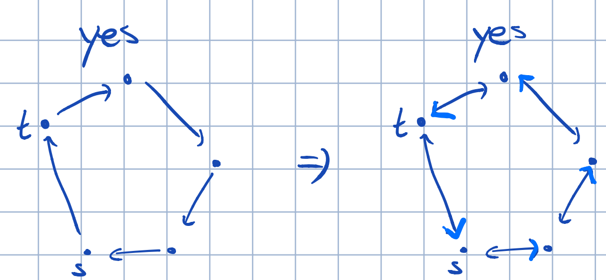
works, because each vertice in a Ham Cycle has exactly one incoming and one outgoing edge.
$\Rightarrow$ after f each edge will be undirected incorrect

$w \in L_1 \Leftarrow f(w) \in L_2$

works, because if we have a undirected Ham Cycle we had an directed Ham Cycle, befor we used f. (else not all edges would be undirected) not according to your f

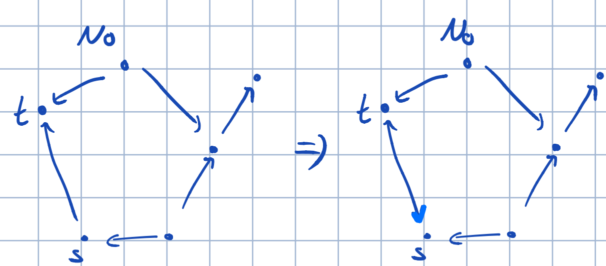
e.g. is there a Ham Cycle? $\Leftrightarrow$ is there a undirected Ham Cycle

1. yes-instant



$\Rightarrow$ yes

2. No-instant



$\Rightarrow$ no

1

(d) TRAVELING SALESPERSON
Input: n cities and pairwise distances d_{ij} (arbitrary non-negative integers), value N.
Output: *yes* if there is a tour visiting each city exactly once, with total distance at most N, *no* otherwise.

L_1 $\qquad$ L_2

Ham Cycle $\leq_p$ Traveling Salesperson

Input Graph $G(E,V)$

f:

- create a full graph $G_2(E_2,V_2)$ with distances $= \begin{cases} 1 & e \in E \\ 2 & e \notin E \end{cases}$
- $n = V_2$, $d_{ij} = E_2$, $N = |V_2|$

$w \in L_1 \overset{?}{\Rightarrow} f(w) \in L_2$

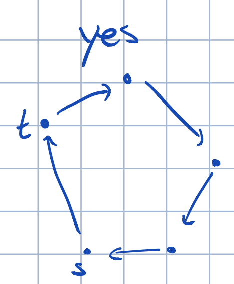
works, because if there is a Ham Cycle, there is a path throug all cities with each edge cost 1. (total N)

$w \in L_1 \Leftarrow f(w) \in L_2$

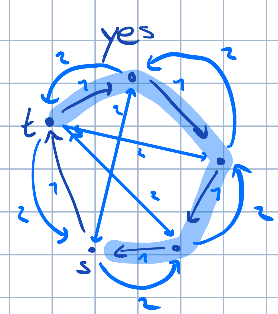
works, because if we have a tour visiting each city exactly once with total distance N, exactly that tour is our Ham cycle (only the last not used 1 distance edge is missing)

e.g. is there a Ham Cycle ? $\Leftrightarrow$ is there a Traveling Salesperson

1. yes-instant

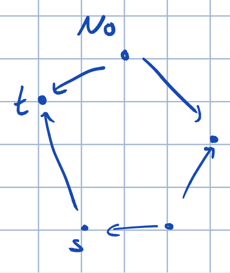


$\Rightarrow$

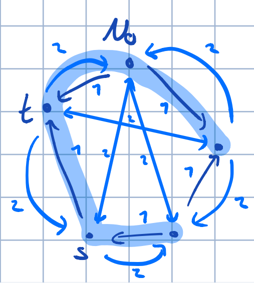


Shortest path = 4 = cities
$\Rightarrow$ yes

2. No-instant



$\Rightarrow$



Shortest path = 5 $\neq$ cities
$\Rightarrow$ no

2

(e) Shortest simple path

Input: graph G with edge weights (possibly negative; negative cycles also allowed), vertices s and t, and a value N.

Output: *yes* if there is an s-t path of length at most N, visiting each vertex *at most once*, no otherwise.

$$\underbrace{\text{Ham path}}_{L_1} \leq_p \underbrace{\text{Shortest simple path}}_{L_2}$$

Input: Graph $G(E,V)$, ~~s, t~~ you need to choose s and t in f

f:

- create a full graph $G_2(E_2, V_2)$ with distances $= \begin{cases} -1 & e \in E \\ \infty & e \notin E \end{cases}$ ✓
- $N = -|V_2|$ ✓

$w \in L_1 \overset{?}{\Rightarrow} f(w) \in L_2$

works, because a Ham path has a $|V_2|$ long simple path and f changes the edge weight of that path to a total $-|V_2|$ ✓

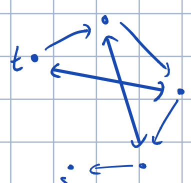
$w \in L_1 \Leftarrow f(w) \in L_2$

works, because if we have a shortest simple path from s to t with value $-|V_2|$. We took exactly $|V_2|$ many edges from s to t (with each vertice exactly once) which is our Ham path ✓

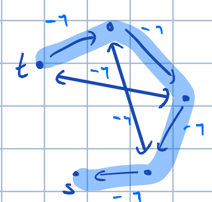
- There cannot be a path with value $\leq N$ $(-|V_2|)$ as there are only $|V_2|$ many -1 edges
- $\geq N$ $(-|V_2|)$ paths are not allowed by def. ✓

e.g. is there a Ham Path ? $\Leftrightarrow$ is there a shortest simple path ?

1. yes-instant



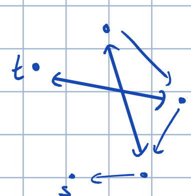
$\Rightarrow$



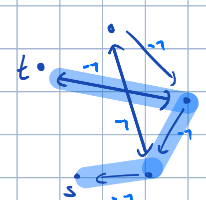
SSP of value $= -|V_2|$

$\Rightarrow$ yes

2. No-instant



$\Rightarrow$



SSP of value $\geq -|V_2|$

$\Rightarrow$ No

1

②

NO.
DATE

Ex2: (a). 3 SAT NP-complete $\Rightarrow$ 4 SAT NP-complete

Firstly, 4SAT is NP problem since all SAT problems are.

WTS 4SAT is NP hard.

Let $k_1 \wedge k_2 \wedge \cdots \wedge k_m$ be an input of 4SAT

where $k_i = l_{i_1} \vee l_{i_2} \vee l_{i_3} \vee l_{i_4}$.

~~$k_i \wedge k_{i+1} = (l_{i_1} \vee l_{i_2} \vee l_{i_3} \vee l_{i_4}) \wedge (l_{i+1,1} \vee l_{i+1,2} \vee l_{i+1,3} \vee l_{i+1,4})$~~

~~$= [(l_{i_1} \vee l_{i_2} \vee l_{i_3}) \wedge (l_{i+1,1} \vee l_{i+1,2} \vee l_{i+1,3})] \vee [(0 \vee 0 \vee l_{i_4})$~~

~~$\wedge (l_{i+1,1} \vee l_{i+1,2} \vee l_{i+1,3})] \vee [(l_{i_1} \vee l_{i_2} \vee l_{i_3}) \wedge (0 \vee 0 \vee l_{i+1,4})]$~~

~~$\vee [(l_{i_1} \vee l_{i_2} \vee l_{i_3}) \wedge (0 \vee 0 \vee l_{i_4}) \wedge (0 \vee 0 \vee l_{i+1,4})]$~~

$k_i = (l_{i_1} \vee l_{i_2} \vee a) \wedge (l_{i_3} \vee l_{i_4} \vee \bar{a})$.

Then ~~4S~~ $\exists f$ function of polynomial time w.r.t. m.

$f: 3SAT \to 4SAT$. this is a reduction from 4SAT to 3SAT

so 4SAT is NP hard since 3SAT is. $\left(\begin{matrix} 4SAT \le_p 3SAT \\ f: 4SAT \to 3SAT \end{matrix} \right)$

$\Rightarrow$ 4SAT is NP complete. you needed the other direction 2

(b) 3 Color (NP complet) $\Rightarrow$ 4 Color. NP complete.

Firstly, 4 Color problem is NP.

certificate: 4 colors and its assignment to vertices.

verifier: check if # colors ≤ 4

and verify every node having different color with its neighbors

WTS 4 Color is NP-hard

$f: G \longrightarrow G'$

3-Colorable graph. (G' graph: $G \vee \{*\}$ is G with an additional node $\{*\}$ connecting to all nodes of G)

if G is 3 colorable $\Leftrightarrow$ G' is four colorable w/ $\{*\}$ the ~~the~~ fourth color. and f is of polynomial time

$\Rightarrow$ 4 color is NP-complete.

3

NO.
DATE

Ex2 (c) Let $K = k_1 \wedge k_2 \wedge \cdots \wedge k_m$ be an input of 2SAT
where $k_i = (a_i \vee b_i)$.

~~$i=1, \ldots, m$~~

~~assign $a_i = 1$~~ ~~search $\bar{a_i}$ in K~~ (*)
~~if $\bar{a_i}$ doesn't exist in K, continue to $i \leftarrow i+1$~~
~~else say $k_{j_1}, \ldots, k_{j_l}$ contain $\bar{a_i}$~~
~~if any of $b_{j_1}, \ldots, b_{j_l}$ was assigned and is 0.~~
~~assign $a_i = 0$.~~
~~else assign and do (*)~~

$i = 1, \ldots, m$.

If a_i hasn't been assigned
assign $a_i = 1$.

(*0)

search $\bar{a_i}$ in K.
if $\bar{a_i}$ doesn't exist in K, continue to $i \leftarrow i+1$
else say $k_{j_1}, \ldots, k_{j_l}$ contain $\bar{a_i}$
If any of $b_{j_1}, \ldots, b_{j_l}$ was assigned and is 0

(*1) is " $\boxed{}$ "
with 0 replaced by 1.
$\bar{a_i}$ replaced by a_i.

assign $a_i = 0$. and do (*1)
if any of $b_{j_1}, \ldots b_{j_l} = 0$
return not satisfiable
else continue

else $b_{j_1} = \cdots = b_{j_l} = 1$.

what if b_{ji} were all unassigned, but $b_{jc} = \overline{b_{jd}}$ for some $i \le c < d \le l$?

else $i \leftarrow i+1$

return value of a_i and b_i for all i.

this is polynomial time since for $\forall\ i = 1, \ldots, m$. the running time is $O(m)$. $\rightsquigarrow$ $O(m^2)$ in total.

2

3

~~(c) let $K = k_1 \wedge k_2 \wedge \cdots \wedge k_m$ be an input of 2SAT.~~
~~where $k_i = (a_i \vee b_i)$~~
~~assign $a_1 = 1$ then search in K of $\overline{a_1}$ in K.~~
~~$i = 1, \cdots, m$.~~
~~assign $a_i = 1$, search $\overline{a_i}$ in K. ~~if $\overline{a_i}$~~~~
~~if $\overline{a_i}$ doesn't exist, continue to $i = i+1$~~
~~else say $k_{j_1}, \cdots, k_{j_L}$ contain $\overline{a_i}$, then assign $b_{j_1}, \cdots, b_{j_L} = 1$.~~

Ex3 (a) If $T \subseteq V$ is an idependent set, then $V \setminus T$ hits all edges.
Suppose $V \setminus T$ doesn't hit all edges. Say $e \in E$ an edge hasn't been hit by $V \setminus T$. it means the 2 ends of e are both not in $V \setminus T$, so both of the ends are in T. then two nodes in T have an edge e conecting them. contraditory to T being independent set. 2

(b). Firstly VC (vertex cover problem) is NP
certificate : $V' \subseteq V$
verify : $V \setminus V'$ independent set or not.
So VC is NP
WTS VC is NP-hard.

polynomial reduction.
For each node (i) of graph G, let x_i be a variable associating to (i), for each edge (v, w), let it associate to claus $x_v \vee x_w$.
a vertex cover of size k corresponds to $x_1 + \cdots + x_n \leq k$.
and CNF is $K = \bigwedge_{e \in E} (x_{e_1} \vee x_{e_2})$. this would be 2SAT ∈ P
$e = (e_1, e_2) \in E$

if K is satisfiable, VC of size k exists. but SAT doesn't care how many variables are set to 1
since SAT is NP hard, VC is also NP-hard.
2