Graph Algorithms - Lecture 10

December 13, 2024

Suppose that in the flow network $R = (G, s, t, c)$, an additional cost function is given: $a : E \to \mathbb{R}$; $\forall ij \in E$, $a(ij) = a_{ij}$ is the cost of arc ij (interpreted as the cost of sending of an "unit" of flow on the arc ij).

If x is a flow in R, then the cost of x is

$$\mathrm{a(x)} = \sum_{\mathrm{i,j}} \mathrm{a_{ij} x_{ij}}.$$

Minimum cost flow problem

Given: R a flow network, $a : E \to \mathbb{R}$ a cost function, and $v \in \mathbb{R}_+$,

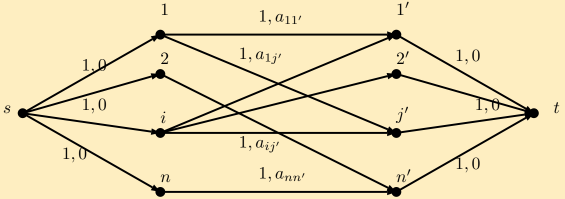
Find: a flow x^0 in R such that

$$a(x^0) = \min \{a(x) \ : \ x \text{ flow in } R, v(x) = v\}.$$

Note that, if v is not greater than the maximum flow value in R, then the problem has always solutions ($a(x)$ is a linear function defined on the non-empty compact set in $\mathbb{R}^{m+1}$ of all flows of value v).

1. Assignment Problem. There are n workers and n jobs. The cost of assigning worker i to job j is a_{ij}. Assign each worker to a job such that the total cost is minimized.

Let us consider the bipartite network flow depicted below, where each arc is labeled with its capacity followed by its cost. Hence, $c_{ij'} = 1$, $c_{si} = 1$, $a_{si} = 0$, $c_{j't} = 1$, and $a_{j't} = 0$, $\forall i, j \in \{1, 2, \ldots, n\}$.



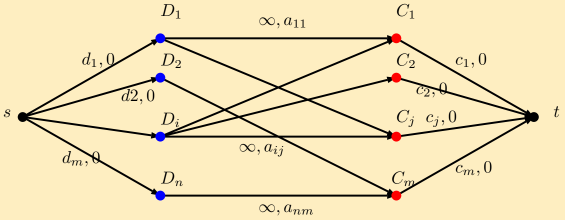
A minimum cost integer flow of value n is a solution of the problem.

Similarly, we can find a perfect matching of minimum weight in a bipartite graph.

2. Hitchcock-Koopmans transportation problem. A commodity, available in the depots $D_1, \ldots, D_n$ in quantities $d_1, \ldots, d_n$, is demanded by customers $C_1, \ldots, C_m$ in quantities $c_1, \ldots, c_m$. The unit transportation cost, a_{ij} - from the depot D_i to the customer C_j, $\forall i \in \{1, \ldots, n\}$,$\forall j \in \{1, \ldots, m\}$ - is known.

Find a transportation schedule satisfying all customers demands and having a minimum total transportation cost.

The problem has a solution only if $\sum_{i=1}^{n} d_i \geqslant \sum_{j=1}^{m} c_j$. In this case, a minimum cost flow of value $v = \sum_{i=1}^{m} c_i$, in the network flow below, solves the problem.



Definition

Let x be a flow in $R = (G, s, t, c)$ and $a : E \to \mathbb{R}$ a cost function.

- *If P is an A-path in R w.r.t. x, then the cost of the path P is defined as*

$$a(P) = \sum_{ij \in E(P), ij \ forward} a_{ij} - \sum_{ij \in E(P), ji \ backward} a_{ji}$$

- *If C is a closed A-path in R w.r.t. x, then $a(C)$ is computed using the above equation, after establishing a direction of traversal of C (it is possible that C is an A-path w.r.t. x in both directions).*

Remarks

- If P is an augmenting path w.r.t. x, then $x^1 = x \otimes r(P)$ is a flow of value $v(x^1) = v(x) + r(P)$ and cost $a(x^1) = a(x) + r(P) \cdot a(P)$.

Remarks

- If C is a closed A-path in R w.r.t. x, then $x^1 = x \otimes r(C)$ is a flow of value $v(x^1) = v(x)$ and cost $a(x^1) = a(x) + r(C) \cdot a(C)$. It follows that if $a(C) < 0$ then x^1 is a flow of the same value as x but of cost $a(x^1) < a(x)$.

Theorem 1

A flow x of value v is a minimum cost flow if and only if there is no negative cost closed A-path w.r.t. x in R.

Proof. "$\Rightarrow$" It follows from the above remark.
"$\Leftarrow$" Let x be a flow (of value v) such that there is no negative cost closed A-path w.r.t. x in R. Let x^* be a minimum cost flow of value v such that

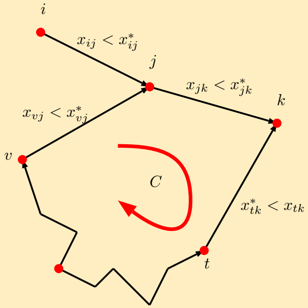
$$\Delta(x, x^*) = \min\{\Delta(x, x') \ : \ x' \text{ minimum cost flow of value } v\},$$

where $\Delta(x, x') = |\{ij \in E \ : \ x_{ij} \neq x'_{ij}\}|$.

If $\Delta(x, x^*) = 0$, then $x = x^*$, hence x is a minimum cost flow.

Otherwise, $\Delta(x, x^*) > 0$ and there is ij such that $x_{ij} \neq x^*_{ij}$. Suppose $0 \leqslant x_{ij} < x^*_{ij} \leqslant c_{ij}$ (a similar argument works if $x_{ij} > x^*_{ij}$). By the flow conservation law, there exists an arc $jk \in E$ such that $0 \leqslant x_{jk} < x^*_{jk} \leqslant c_{jk}$, or there is $kj \in E$ such that $0 \leqslant x^*_{kj} < x_{kj} \leqslant c_{kj}$.

Since the number of vertices is finite, by repeating this argument, we construct, C, a closed A-path w.r.t. x in R:



If we traverse C in the converse direction, we obtain a closed A-path, C', w.r.t. x^*. Since $a(C) \geqslant 0$ (by the hypothesis), and $a(C') = -a(C)$ it follows that $a(C) = 0$. (x^* is of minimum cost; hence, by the first part of the proof, $a(C') \geqslant 0$).

If we consider $x' = x^* \otimes \delta(C')$, where

$$\delta(C') = \min\left\{\min_{kj \textit{ forward in } C'}(x_{kj} - x^*_{kj}), \min_{kj \textit{ backward in } C'}(x^*_{jk} - x_{jk})\right\},$$

then x' satisfies $v(x') = v(x^*) = v$, $a(x') = a(x^*) + \delta(C') \cdot a(C') = a(x^*)$.

Hence x' is a minimum cost flow of value v, but $\Delta(x, x') < \Delta(x, x^*)$, contradicting the choice of x^*. Thus $\Delta(x, x^*) = 0$, and the theorem is proved. $\square$

Theorem 2

If x is a minimum cost flow of value v and P_0 is an augmenting path w.r.t. x such that

$$a(P_0) = \min\{a(P) \,:\, P \text{ augmenting path w.r.t. } x\},$$

then $x^1 = x \otimes r(P_0)$ is a minimum cost flow of value $v(x^1) = v + r(P_0)$.

Proof. Omitted.

An augmenting path of minimum cost can be found using shortest paths algorithms. If x is a flow in R and $a : E \to \mathbb{R}$ is the cost function, then taking $a_{ij} = \infty$ if $ij \notin E$ (when $x_{ij} = 0$), we define

$$\overline{a}_{ij} = \begin{cases} a_{ij}, & \text{if } x_{ij} < c_{ij} \text{ and } x_{ji} = 0, \\ \min\{a_{ij}, -a_{ji}\}, & \text{if } x_{ij} < c_{ij} \text{ and } x_{ji} > 0, \\ -a_{ji}, & \text{if } x_{ij} = c_{ij} \text{ and } x_{ji} > 0, \\ +\infty, & \text{if } x_{ij} = c_{ij} \text{ and } x_{ji} = 0. \end{cases}$$

A shortest $\overline{a}$ st-path corresponds to a minimum cost augmenting path w.r.t. x in R, and a negative a cycle corresponds to a negative cost closed A-path w.r.t. x in R.

Then, we have the following algorithm to solve the minimum cost flow problem:

let x be a flow of value $v' \leqslant v$;
// x could be null or $x = (v/v(y) \cdot y$,
// where y is a flow of maximum value.
while ($\exists C$ an $\overline{a}$ negative cycle) do
    $x \leftarrow x \otimes r(C)$;
end while
while $(v(x) < v)$ do
    find an shortest $\overline{a}$ st-path P;
    $x \leftarrow x \otimes \min\{r(P), v - v(x)\}$;
end while

The time complexity of the second while is $\mathcal{O}(n^3 v)$ (if we start with the null flow and if we have integer capacities only). The first while can be implemented to run in $\mathcal{O}(nm^2 \log n)$ iterations.

Reminder

- Let $P_i : I_i \to \{yes, no\}$ $(i \in \{1, 2\})$ be two decision problems. P_1 polynomial reduces to P_2, and we denote this by $P_1 \leqslant_P P_2$, if there is a polynomial-time computable function $\Phi : I_1 \to I_2$, such that $P_1(i) = P_2(\Phi(i))$, $\forall i \in I_1$.
- The function Φ will be given using an algorithm that constructs, for every instance $i_1 \in I_1$, an instance $i_2 \in I_2$ in polynomial time (in the size of i_1), such that $P_1(i_1) = yes$ if and only if $P_2(i_2) = yes$.
- The construction behind a polynomial-time reduction shows how the first problem can be efficiently solved using an oracle for the second one.

- The relation $\leqslant_P$ is a transitive relation on the set of decision problems (since, the class of polynomial functions is closed to function composition).

Example

SAT $\leqslant_P$ 3SAT

SAT

Instance: $U = \{u_1, u_2, \ldots, u_n\}$ a finite set of boolean variables;

$$C = C_1 \wedge C_2 \ldots \wedge C_m \text{ a CNF formula over } U:$$

$$C_i = v_{i1} \vee v_{i_2} \vee \ldots \vee v_{i_{k_i}}, i = \overline{1, m}, \text{ where}$$

$$\forall i_j, \exists \alpha \in \{1, 2, \ldots, n\} \text{ s. t. } v_{i_j} = u_\alpha \text{ or } v_{i_j} = \overline{u}_\alpha.$$

Question: Is there an assignment $t : U \to \{true, false\}$ s. t. $t(C) = true$?

3SAT is the restriction of SAT to the set of instances in which each clause C_i has exactly 3 literals ($k_i = 3$), where a literal, v_{i_j}, as described above, is a variable or its negation.

The SAT problem is famous since it was the first discovered NP-complete problem (Cook, 1971).

$$\mathrm{NP} \neq \mathrm{NP} \cap \mathrm{coNP} = \mathrm{P}$$

SM

Instance: $G = (V, E)$ a graph and $k \in \mathbb{N}$.

Question: Is there a stable set S in G such that $|S| \geqslant k$?

Theorem 3

(Karp, 1972). 3SAT $\leqslant_P$ SM.

Proof. Let $U = \{u_1, u_2, \ldots, u_n\}$, $(n \in \mathbb{N}^*)$, $C = C_1 \wedge C_2 \ldots \wedge C_m$ $(m \in \mathbb{N}^*)$ with $C_i = v_{i1} \vee v_{i_2} \vee v_{i_3}$, $i = \overline{1, m}$, (where $\forall i_j, \exists \alpha \in \{1, 2, \ldots, n\}$ s. t. $v_{i_j} = u_\alpha$ or $v_{i_j} = \overline{u}_\alpha$) representing the data of an instance of the 3SAT problem.

We will build in $\mathcal{O}(n + m)$ time complexity, a graph G and $k \in \mathbb{N}$, such that there is a satisfying assignment t for C if and only if there is a stable set S in G such that $|S| \geqslant k$.

Proof cont'd.

Construction of the graph G:

- $\forall i \in \{1, 2, \ldots, n\}$, let the disjoint graphs $T_i = (\{u_i, \overline{u}_i\}, \{u_i\overline{u}_i\})$.
- $\forall j \in \{1, 2, \ldots, m\}$, let the disjoint graphs $Z_j = (\{a_{j1}, a_{j2}, a_{j3}\}, \{a_{j1}a_{j2}, a_{j2}a_{j3}, a_{j3}a_{j1}\})$.
- $\forall j \in \{1, 2, \ldots, m\}$, let the edge-set $E_j = \{a_{j1}v_{j1}, a_{j2}v_{j2}, a_{j3}v_{j3}\}$, where $v_{j1} \vee v_{j2} \vee v_{j3}$ is the clause C_j.

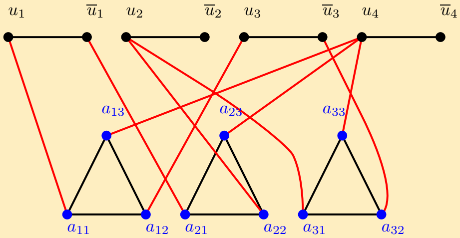
$$V(G) = \left(\bigcup_{i=1}^{n} V(T_i)\right) \cup \left(\bigcup_{j=1}^{m} V(Z_j)\right)$$

$$E(G) = \left(\bigcup_{i=1}^{n} E(T_i)\right) \cup \left(\bigcup_{j=1}^{m} E(Z_j) \cup E_j\right).$$

Clearly, the construction of G can be done in polynomial time w.r.t. the size $n+m$ of the 3SAT instance. Let $k=n+m$.

Example

$U=\{u_1,u_2,u_3,u_4\}$; $C=(u_1\vee u_3\vee u_4)\wedge(\overline{u}_1\vee u_2\vee u_4)\wedge(u_2\vee\overline{u}_3\vee u_4)$; $k=4+3=7$.



Suppose that the answer of the problem SM for the above (G, k) instance is yes.

So, $\exists S \in \mathcal{S}_G$ (the family of all stable sets in G) such that $|S| \geqslant k$. Since any stable set can have at most one vertex from each $V(T_i)$ and each $V(Z_j)$, it follows that $|S| = k$ and $|S \cap V(T_i)| = 1$, $|S \cap V(Z_j)| = 1$, $\forall i = \overline{1, n}$, $\forall j = \overline{1, m}$.

Let $t : U \to \{true, false\}$ given by

$$t(u_i) = \begin{cases} true, & \text{if } S \cap V(T_i) = \{\overline{u}_i\} \\ false, & \text{if } S \cap V(T_i) = \{u_i\} \end{cases}$$

Then, $t(C_j) = true$, $\forall j = \overline{1, m}$, hence $t(C) = true$, and the answer of 3SAT is yes.

Indeed, $\forall j = \overline{1, m}$, if $C_j = v_{j1} \vee v_{j2} \vee v_{j3}$ and $S \cap V(Z_j) = \{a_{jk}\}$ ($k \in \{1, 2, 3\}$), then (since $a_{jk}v_{jk} \in E$) it follows that $v_{jk} \notin S$.

- If $v_{jk} = u_\alpha$, then $u_\alpha \notin S$, so $\overline{u}_\alpha \in S$, and - by the definition of t - we have $t(u_\alpha) = true$, that is, $t(v_{jk}) = true$, which implies $t(C_j) = true$.
- If $v_{jk} = \overline{u}_\alpha$, then $\overline{u}_\alpha \notin S$, so $u_\alpha \in S$, and - by the definition of t - we have $t(\overline{u}_\alpha) = true$, that is, $t(v_{jk}) = true$, which implies $t(C_j) = true$.

Conversely, if the answer of the 3SAT is yes, then $\exists t : U \to \{true, false\}$ such that $t(C_j) = true$, $\forall j = \overline{1, m}$.

Let $\overline{S}_1$ be the stable set $\overline{S}_1 = \bigcup_{i=1}^{n} V'_i$ with n vertices, where

$$V'_i = \begin{cases} \{\overline{u}_i\}, & \text{if } t(u_i) = true\} \\ \{u_i\}, & \text{if } t(u_i) = false\} \end{cases}$$

Then, since $t(C_j) = true, \forall j = \overline{1, m}$, it follows that there is $k_j \in \{1, 2, 3\}$ such that $t(v_{jk_j}) = true$. Let $\overline{S}_2 = \bigcup_{j=1}^{m} \{a_{jk_j}\}$. Obviously, $\overline{S}_2$ is a stable set in G having m vertices.

Let $\overline{S} = \overline{S}_1 \cup \overline{S}_2$. Obviously, $|\overline{S}| = n + m = k$ (hence $|\overline{S}| \geqslant k$). If we prove that $\overline{S}$ is a stable set, then the answer of SM for the instance (G, k) is yes.

Suppose that $\exists v, w \in \overline{S}$ such that $e = vw \in E(G)$. Then one extremity of e is in $\overline{S}_1$ and the other in $\overline{S}_2$. If $v \in \overline{S}_1$, then we have two cases:

- $v = u_\alpha$, $w = a_{jk_j}$, $\alpha \in \{1, 2, \ldots, n\}$, $j \in \{1, 2, \ldots, m\}$, $k_j \in \{1, 2, 3\}$ and $v_{jk_j} = u_\alpha$.
  Since $t(v_{jk_j}) = true$, it follows that $t(u_\alpha) = true$, therefore $v = u_\alpha \notin \overline{S}_1$, contradiction.

- $v = \overline{u}_\alpha$, $w = a_{jk_j}$, $\alpha \in \{1, 2, \ldots, n\}$, $j \in \{1, 2, \ldots, m\}$, $k_j \in \{1, 2, 3\}$ and $v_{jk_j} = \overline{u}_\alpha$.
  Since $t(v_{jk_j}) = true$, it follows that $t(\overline{u}_\alpha) = true$, therefore $t(u_\alpha) = false$. Hence, $v = \overline{u}_\alpha \notin \overline{S}_1$, contradiction. $\square$

Note that a similar proof can be given to prove SAT $\leqslant_P$ SM, the only difference is that the graphs Z_i are complete graphs with k_i vertices.

COL

Instance: $G = (V, E)$ graph and $k \in \mathbb{N}^*$.

Question: Is there a k-coloring of (the vertices) of G?

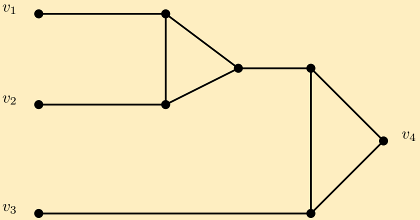
Theorem 4

3SAT $\leqslant_P$ COL.

This theorem shows that the vertex coloring problem is NP-hard. The proof given below shows more: the problem, obtained from COL by restricting only to instances with $k = 3$, is NP-hard, too.

Lemma 1

Let H be the graph

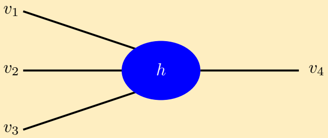


a) If c is a 3-coloring of H s. t. $c(v_1) = c(v_2) = c(v_3) = a \in \{1, 2, 3\}$, then necessarily $c(v_4) = a$ (forcing).

b) If $c : \{v_1, v_2, v_3\} \to \{1, 2, 3\}$ satisfies $c(\{v_1, v_2, v_3\}) \neq \{a\}$, ($a \in \{1, 2, 3\}$), then c can be extended to a 3-coloring c of H with $c(v_4) \neq a$.

Proof. By examining the list of 3-colorings of H.

We will use the following simplified representation of the graph H:

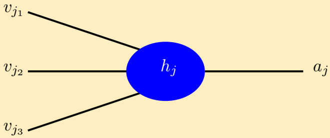


Proof of Theorem 4. Let us consider the data of an instance of 3SAT: $U = \{u_1, \ldots, u_n\}$, $(n \in \mathbb{N}^*)$, a set of boolean variables, and $C = C_1 \wedge \ldots \wedge C_m$, $(m \in \mathbb{N}^*)$ a CNF-formula with $C_j = v_{j_1} \vee v_{j_2} \vee v_{j_3}$, $\forall j = \overline{1, m}$, where $\forall i = \overline{1, 3}$, $\exists \alpha$ s. t. $v_{j_i} = u_\alpha$ or $v_{j_i} = \overline{u}_\alpha$.

We will build a graph G with the property that G is 3-colorable if and only if the answer of 3SAT for this instance is yes, that is, there is an assignment $t : U \to \{\mathit{true}, \mathit{false}\}$ such that $t(C) = \mathit{true}$.

The construction needs polynomial time, and consists in the following steps:

- Consider the disjoint graphs (V_i, E_i), $\forall i = \overline{1, n}$, where $V_i = \{u_i, \overline{u}_i\}$ and $E_i = \{u_i\overline{u}_i\}$.
- For $C_j = v_{j_1} \vee v_{j_2} \vee v_{j_3}$, $\forall j = \overline{1, m}$, consider the graphs:

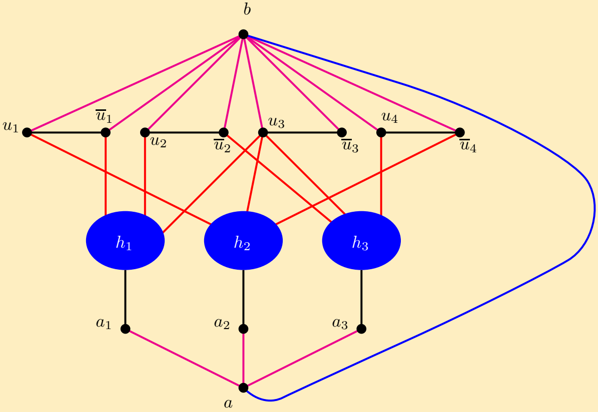


  where, v_{j_k} $(k = \overline{1, 3})$ are the vertices corresponding to the literals v_{j_k}, the graphs h_j are disjoint, and a_j are distinct vertices.
- Consider a new vertex a, and all edges aa_j, $\forall j = \overline{1, m}$.
- Consider a new vertex b, all edges bu_i, $b\overline{u}_i$, $\forall i = \overline{1, n}$ and ba.

The graph G has a linear number of vertices w.r.t. $n + m$.

Example

$U = \{u_1, u_2, u_3, u_4\}$, $C = (\overline{u}_1 \vee u_2 \vee u_3) \wedge (u_1 \vee u_3 \vee \overline{u}_4) \wedge (\overline{u}_2 \vee u_3 \vee u_4)$.
The graph G is



Suppose that the answer of 3SAT for the considered instance is yes

Hence $\exists t : U \to \{true, false\}$ s. t. $t(C) = true$, that is, $t(C_j) = true$, $\forall j = \overline{1, m}$. We will show that G is 3-colorable.

We color, firstly, vertices u_i and $\overline{u}_i$, $\forall i \overline{1, n}$.

$$\begin{cases} c(u_i) = 1 \text{ and } c(\overline{u}_i) = 2, \text{if } t(u_i) = true \\ c(u_i) = 2 \text{ and } c(\overline{u}_i) = 1, \text{if } t(u_i) = false \end{cases}$$

Note that, if v is a literal, then $c(v) = 2$ if and only if $t(v) = false$.

Since t is a satisfying assignment, $t(C_j) = true$, $\forall j = \overline{1, m}$. It follows that $c(\{v_{j_1}, v_{j_2}, v_{j_3}\}) \neq \{2\}$, $\forall j = \overline{1, m}$.

By Lemma 1 b), we can extend c to a 3-coloring, in each graph h_j, such that $c(a_j) \neq 2$, that is $c(a_j) \in \{1, 3\}$.

By taking $c(a) = 2$ and $c(b) = 3$, we obtain a 3-coloring of G.

Conversely, suppose that G is 3-colorable.

We can assume that $c(b) = 3$ and $c(a) = 2$ (otherwise, rename the colors).

It follows that $\{c(u_i), c(\overline{u}_i)\} = \{1, 2\}$, $\forall i = \overline{1, n}$ and $c(a_j) \in \{1, 3\}$, $\forall j = \overline{1, m}$.

By Lemma 1 a), it follows that $c(\{v_{j_1}, v_{j_2}, v_{j_3}\}) \neq 2$, $\forall j = \overline{1, m}$. This means that, $\forall j = \overline{1, m}$, there is a $v_{j_k} \in C_j$ such that $c(v_{j_k}) = 1$.

Hence, defining $t : U \to \{true, false\}$ by

$$t(u_i) = \begin{cases} true, & \text{if } c(u_i) = 1 \\ false, & \text{if } c(u_i) = 2 \end{cases},$$

we obtain an assignment with the property that $t(C_j) = true$, $\forall j = \overline{1, m}$.

Therefore the answer of 3SAT for the given instance is yes.

Remember: Let G be a (di)graph. A cycle C of G is a Hamiltonian cycle if $V(C) = V(G)$.
On open path P of G is a Hamiltonian path if $V(P) = V(G)$. A Hamiltonian (di)graph is a (di)graph which has a Hamiltonian cycle. A traceable (di)graph is a (di)graph which has a Hamiltonian path.

Theorem 5

(Nash-Williams, 1969) The following five problems are polynomial equivalent:

CH: Given a graph G. Is G Hamiltonian?

TR: Given a graph G. Is G traceable?

DCH: Given a digraph G. Is G Hamiltonian?

DTR Given a digraph G. Is G traceable?

BCH: Given a bipartite graph G. Is G Hamiltonian?

Remark

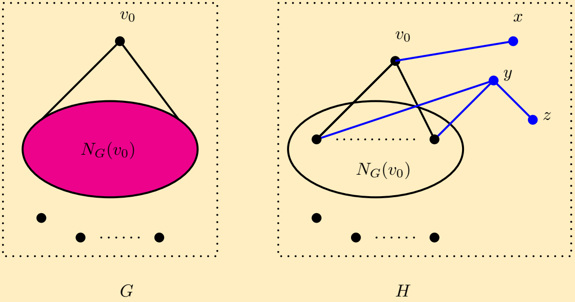
P_1 and P_2 are polynomial equivalent if $P_1 \leqslant_P P_2$ and $P_2 \leqslant_P P_1$.

Proof of Theorem 5.

CH $\leqslant_P$ TR Let G be a graph and $v_0 \in V(G)$. We construct in polynomial time a graph H such that G is Hamiltonian if and only if H is traceable.

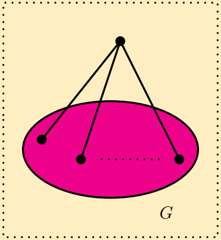
Let $V(H) = V(G) \cup \{x, y, z\}$ and $E(H) = E(G) \cup \{xv_0, yz\} \cup \{wy : w \in V(G) \cap N_G(v_0)\}$.

Proof of Theorem 5 cont'd.



Then, H is traceable if and only if it has a Hamiltonian path, P, with extremities x and z (which have degree 1 in H). P exists in H if and only if in G there is a Hamiltonian path with an extremity in v_0 and the other a neighbor of v_0, that is, if and only if G is Hamiltonian.

TR $\leqslant_P$ CH Let G be a graph. Consider $H = G + K_1$. Then, H is Hamiltonian if and only if G has a Hamiltonian path.



$G + K_1$

The equivalence of the problems DCH and DTR can be proved in a similar way.

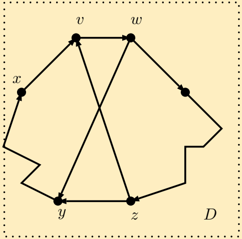
CH $\leqslant_P$ DCH Let G be a graph. Let D be the digraph obtained from G by replacing each edge with a symmetric pair of arcs.

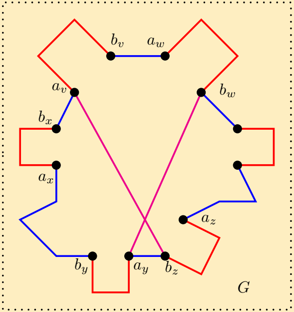
Clearly any cycle in G gives rise to a cycle in D and conversely, any cycle in D gives rise to a cycle in G.

DCH $\leqslant_P$ CH Let D be a digraph. Each vertex $v \in V(D)$ is replaced by an undirected graph $P_3(v)$ with extremities a_v and b_v:

$$P_3(v) = (\{a_v, b_v, c_v, d_v\}, \{a_v c_v, c_v d_v, d_v b_v\}).$$

Each arc $vw \in E(D)$ is replaced by the (undirected) edge $b_v a_w$. Let G be the graph obtained (in polynomial time) in this way:





Each cycle C of D corresponds to a cycle in G and conversely, each cycle in G corresponds to a cycle of D. It follows that D is Hamiltonian if and only if G is Hamiltonian.

Note that if C is a cycle of G, then this is generated by a cycle C' of D, and $length(C) = 3 \cdot length(C') + length(C') = 4 \cdot length(C')$. It follows that any cycle of G is even, therefore G is a bipartite graph.

Therefore the above proof (DCH $\leqslant_P$ CH) is in fact DCH $\leqslant_P$ BCH.

Since BCH $\leqslant_P$ CH is obvious, the Theorem 5 is completely proved. $\square$

Memento

- In order to prove that a certain decision problem is NP-complete:

I. First you have to show that the problem is in the class NP which means: there exists a polynomial time certificate for a any solution-candidate (for 3COL: given a function $c : V(G) \to \mathbb{N}$, we can verify if c is a coloring and uses at most three colors in $\mathcal{O}(n^2)$).

II. Second, we must polynomially reduce a known NP-complete problem to your decision problem (for 3COL: e. g. 3SAT can be polynomially reduced to 3COL).

- If step I. misses, then the decision problem in sight is NP-hard.

Exercise 1. Show that the following problem is NP-complete

INT

Instance: $n, m \in \mathbb{N}^*$, $A \in \mathbb{Z}^{m \times n}$, $b \in \mathbb{Z}^m$.

Question: Is there an assignment $x \in \mathbb{Z}^n$ s. t. $Ax \leqslant b$?

(The inequality $\leqslant$ between two vectors is componentwise.) *Hint:* You can try SM $\leqslant_P$ INT.

Exercise 2. Consider the following decision problem

ACYCLIC

Instance: $G = (V, E)$ a digraph and $p \in \mathbb{N}$.

Question: Is there $A \subseteq V$ such that $|A| \leqslant p$ and $G - A$ doesn't contain cycles?

Prove that SM $\leqslant_P$ ACYCLIC.

Exercise 3. A k-uniform hypergraph is a pair $H = (V, E)$, where $V \neq \varnothing$ is a finite set, $k \in \mathbb{N}^* \setminus \{1\}$, and $E \subseteq \mathcal{P}_k(V) = \{A \subseteq E \ : \ |A| = k\}$. It easy to see that a 2-uniform hypergraph is a simple graph.

We say that a k-uniform hypergraph $H = (V, E)$ is *simple* if there is a function $c : V \to \{1, 2, \ldots, k\}$ such that $\forall u, v \in V$, $u \neq v$, if $u, v \in e$ for a certain $e \in E$, then $c(u) \neq c(v)$. Now we consider the following decision problem

k-SIMPLE

Instance: H a k-uniform hypergraph.

Question: Is H simple?

(a) Prove that problem 3-SIMPLE is NP-complete.

(b) Show that problem 2-SIMPLE belongs to P.

Exercise 4. Consider the following decision problem:

AGM

Instance: G a graph, $k \in \mathbb{N}$.

Question: Has G a spanning tree T such that $\Delta(T) \geqslant k$?

Prove that AGM $\in$ P.

Exercise 5. Let $D = (V, E)$ a loopless digraph. A stable set of D, $S \subseteq V$, is a quasi-kernel if every vertex $v \in V \setminus S$ can be accessed from inside S along a path having length at most 2.

(a) Prove that a quasi-kernel can be constructed in $\mathcal{O}(n + m)$, where $n = |V|$ and $m = |E|$.

(b) Show that 3-SAT is polinomially reducible to the problem of finding out if in a given digraph it exists a quasi-kernel containing a given vertex.

Exercise 6. Consider the following decision problem: LPL

Instance: G a graph, $k \in \mathbb{N}$.

Question: Has G a path P such that $length(P) \geqslant k$?

Prove that LPL is NP-complete.

Exercise 7. A kernel in a digraph $G = (V, E)$ is a stable set $S \subseteq V$ such that $\forall\, u \in V \setminus S$ exists $v \in S$ with $vu \in E$. We consider the following decision problem:

KERNEL

Instance: G a digraph.

Question: Has G a kernel?

Prove that the following construction leads to a polynomial reduction of SAT to KERNEL (i.e. SAT $\leqslant_P$ KERNEL):

Exercise 7 (cont'd). For every conjuction of clauses, F, instance of SAT, we define a digraph G (an instance for NUCLEU):

- for each clause C of F we add a 3-cycle to G

$$v_C^1, v_C^1 v_C^2, v_C^2, v_C^2 v_C^3, v_C^3, v_C^3 v_C^1;$$

- for every variable x which occurs in formula F, we add a 2-cycle to G

$$v_x, v_x v_{\overline{x}}, v_{\overline{x}}, v_{\overline{x}} v_x, v_x;$$

- for every clause C and every literal u which occurs in C we add to G three arcs

$$v_u v_C^1, v_u v_C^2, v_u v_C^3.$$

Exercises for the 11th seminar

Exercise 8. Consider the following decision problem

2SAT

Instance: $\mathcal{C}$ a family of clauses of exactly two literals each.

Question: There exists an assignment of truth to the variables which satisfies all the clauses form $\mathcal{C}$?

We define G (the implication) digraph: $V(G) =$ set of all literals used in $\mathcal{C}$ and $E(G) = \{\overline{v}_j w_j, \overline{w}_j v_j \ : \ C_j = v_j \vee w_j,\ j = \overline{1, m}\}$ (each clause introduces in G two arcs). Show that $\mathcal{C}$ is satisfiable if and only if x_i and $\overline{x}_i$ belong to different strongly connected components of G, $\forall i = \overline{1, n}$. Show that this property can be tested in $\mathcal{O}(n + m)$ time complexity.

Exercise 9. Show that the following problem is NP-complete.

MAX-2SAT

Instance: $\mathcal{C}$ a family of clauses of at most two literals each and $k \in \mathbb{N}$.

Question: There exists an assignment of truth to the variables which satisfies at least k clauses form $\mathcal{C}$?

Exercise 10. Consider the following decision problem

NAE-3SAT

Instance: $\mathcal{C}$ a family of clauses of exactly three literals each.

Question: Is there a truth assignment to the variables such that each clause has at least one true literal and at least one false literal?

Prove that the following construction leads to a polynomial reduction of 3SAT to NAE 3SAT (i.e. 3SAT $\leqslant_P$ NAE 3SAT):

- we keep the boolean variables from 3SAT, $U = \{u_1, u_2, \ldots, u_n\}$ and add a new variable x;
- for each clause $C_j = v_{j_1} \vee v_{j_2} \vee v_{j_3}$ we add a new variable y_j and we replace C_j by two clauses:

$$C_j^1 = v_{j_1} \vee v_{j_2} \vee y_j, C_j^2 = v_{j_3} \vee x \vee \overline{y}_j.$$

Exercise 11. Consider the following decision problem

MAX-CUT

Instance: $G = (V, E)$ a graph, $c : E \to \mathbb{R}$ a weight on its edges, $k \in \mathbb{R}$.

Question: Is there a cut of G of weight at least k?

Prove that the following construction leads to a polynomial reduction of NAE 3SAT to MAXCUT (i.e. NAE 3SAT $\leqslant_P$ MAXCUT):

- consider an instance for NAE 3SAT problem with clauses $\mathcal{C} = \{C_1, \dots, C_m\}$ over the set of boolean variables $U = \{u_1, u_2, \dots, u_n\}$; we can suppose that every clause contains three different literals,
- $V(G) = \{u_i, \overline{u}_i : i = \overline{1, n}\}$ and we add to $E(G)$ the edges $u_i\overline{u}_i$ of weight $10m$,
- for each clause $C_j = v_{j_1} \vee v_{j_2} \vee v_{j_3}$ we add to $E(G)$ three edges: $v_{j_1}v_{j_2}$, $v_{j_2}v_{j_3}$, and $v_{j_1}v_{j_3}$ each of weight 1,
- $k = 10nm + 2m$.