Complexity Theory
VU 181.142, SS 2016

5. NP-Completeness

Reinhard Pichler

Institut für Informationssysteme
Arbeitsbereich DBAI
Technische Universität Wien

19 April, 2016
Outline

5. NP-Completeness
5.1 Some Variants of Satisfiability
5.2 CIRCUIT SAT
5.3 NOT-ALL-EQUAL-SAT
5.4 1-IN-3-SAT
5.5 Some Graph Problems
5.6 3-COLORABILITY
5.7 HAMILTON-PATH, etc.
5.8 Summary
Some Variants of Satisfiability

We have already encountered several versions of satisfiability problems:

- **intractable**: SAT, 3-SAT
- **tractable**: 2-SAT, HORNSAT
Some Variants of Satisfiability

We have already encountered several versions of satisfiability problems:

- intractable: SAT, 3-SAT
- tractable: 2-SAT, HORNSAT

We shall encounter further intractable versions of satisfiability problems:

- restricted (but still intractable) versions of SAT
- CIRCUIT SAT
- Not-all-equal SAT (NAESAT)
- (MONOTONE) 1-IN-3-SAT
- strongly related problem: HITTING SET
Narrowing NP-complete languages

An NP-complete language can sometimes be narrowed down by transformations which eliminate certain features of the language but still preserve NP-completeness.

Restricting SAT to formulae in CNF and a further restriction to 3-SAT are typical examples. Generally, k-SAT (i.e., formulae are restricted to CNF with exactly $k$ literals in each clause) is NP-complete for any $k \geq 3$. 
Narrowing NP-complete languages

An NP-complete language can sometimes be narrowed down by transformations which eliminate certain features of the language but still preserve NP-completeness.

Restricting SAT to formulae in CNF and a further restriction to 3-SAT are typical examples. Generally, k-SAT (i.e., formulae are restricted to CNF with exactly \( k \) literals in each clause) is NP-complete for any \( k \geq 3 \).

Here is another example of narrowing an NP-complete language:

**Proposition**

3-SAT remains NP-complete even if the Boolean expressions \( \varphi \) in 3-CNF are restricted such that

- each variable appears at most three times in \( \varphi \) and
- each literal appears at most twice in \( \varphi \).
Proof

The reduction consists in rewriting an arbitrary instance $\varphi$ of $3$-$\text{SAT}$ in such a way that the forbidden features are eliminated.

Consider a variable $x$ appearing $k > 3$ times in $\varphi$.

(i) Replace the first occurrence of $x$ in $\varphi$ by $x_1$, the second by $x_2$, and so on where $x_1, \ldots, x_k$ are new variables.

(ii) Add clauses $(\neg x_1 \lor x_2), (\neg x_2 \lor x_3), \ldots, (\neg x_k \lor x_1)$ to $\varphi$.

Let $\varphi'$ be the result of systematically modifying $\varphi$ in this way. Clearly, $\varphi'$ has the desired syntactic properties.

Now $\varphi$ is satisfiable iff $\varphi'$ is satisfiable:
For each $x$ appearing $k > 3$ times in $\varphi$, the truth values of $x_1, \ldots, x_k$ are the same in each truth assignment satisfying $\varphi'$. 
Boolean Circuits

Syntax of Boolean circuits

- A Boolean circuit is a directed graph \( C = (V, E) \) where
  \( V = \{1, 2, \ldots, n\} \) is the set of gates and
  \( C \) is acyclic (with \( i < j \) for all edges \( (i, j) \in E \)).

- All gates \( i \) have a sort \( s(i) \in \{\text{true}, \text{false}, \land, \lor, \neg\} \cup \{x_1, x_2, \ldots\} \).
  - If \( s(i) \in \{\text{true}, \text{false}\} \cup \{x_1, x_2, \ldots\} \), the indegree of \( i \) is 0 (inputs).
  - If \( s(i) = \neg \) then the indegree of \( i \) is 1.
  - If \( s(i) \in \{\lor, \land\} \) then the indegree of \( i \) is 2.

- Gate \( n \) is the output of the circuit.

Remark. \( \{x_1, x_2, \ldots\} \) are variables whose value can be true or false.
Boolean Circuits

Semantics

Let $C$ be a Boolean circuit and let $X(C)$ denote the set of variables appearing in the circuit $C$. A truth assignment for $C$ is a function $T : X(C) \rightarrow \{\text{true}, \text{false}\}$.

The truth value $T(i)$ for each gate $i$ is defined inductively:

- If $s(i) = \text{true}$, $T(i) = \text{true}$ and if $s(i) = \text{false}$, $T(i) = \text{false}$.
- If $s(i) = x_j \in X(C)$, then $T(i) = T(x_j)$.
- If $s(i) = \neg$, then $T(i) = \text{true}$ if $T(j) = \text{false}$, else $T(i) = \text{false}$ where $(j, i)$ is the unique edge entering $i$.
- If $s(i) = \land$, then $T(i) = \text{true}$ if $T(j) = T(j') = \text{true}$ else $T(i) = \text{false}$ where $(j, i)$ and $(j', i)$ are the two edges entering $i$.
- If $s(i) = \lor$, then $T(i) = \text{true}$ if $T(j) = \text{true}$ or $T(j') = \text{true}$ else $T(i) = \text{false}$ where $(j, i)$ and $(j', i)$ are the two edges entering $i$.
- $T(C) = T(n)$, i.e. the value of the circuit $C$. 
CIRCUIT SAT

INSTANCE: Boolean circuit \( C \) with variables \( X(C) \)
QUESTION: Does there exist a truth assignment \( T : X(C) \rightarrow \{ \text{true, false} \} \) such that \( T(C) = \text{true} \)?

Theorem

CIRCUIT SAT is \text{NP-complete}.

Proof of NP-Membership

Consider the following NP-algorithm:

1. Guess a truth assignment \( T : X(C) \rightarrow \{ \text{true, false} \} \).
2. Check that \( T(C) = \text{true} \) holds.
Proof of NP-Hardness

We prove the NP-hardness by a reduction from **SAT**: Let an arbitrary instance of **SAT** be given by a Boolean formula $\varphi$ over the variables $X = \{x_1, \ldots, x_k\}$. We construct the following Boolean circuit $C(\varphi)$:

- **The variables $X(C)$** in $C(\varphi)$ are precisely the variables $X$.
- **For every subexpression $\psi$ of $\varphi$, $C(\varphi)$ contains a gate $g(\psi)$. The output gate of $C(\varphi)$ is the gate $g(\varphi)$.**
- **The sort and the incoming arcs of each gate $g(\psi)$ in $C(\varphi)$ are defined inductively:**
  - If $\psi$ is a variable $x_i$ then $g(\psi)$ is an input gate of sort $s(g(\psi)) = x_i$
  - If $\psi = \neg \psi'$ then $s(g(\psi)) = \neg$ with an incoming arc from $g(\psi')$.
  - If $\psi = \psi_1 \land \psi_2$ (resp. $\psi = \psi_1 \lor \psi_2$), then $s(g(\psi)) = \land$ (resp. $s(g(\psi)) = \lor$) with incoming arcs from $g(\psi_1)$ and $g(\psi_2)$. 
Reduction from SAT to 3-SAT

Motivation

- We have already seen how an arbitrary propositional formula $\varphi$ can be transformed efficiently into a sat-equivalent formula $\psi$ in 3-CNF.
- This transformation (first into CNF and then into 3-CNF) is intuitive and clearly works in polynomial time. However, the log-space complexity of this transformation is not immediate.
- We now give an alternative transformation by reducing CIRCUIT SAT to 3-SAT. In total, we thus have:

$$\text{SAT} \leq_L \text{CIRCUIT SAT} \leq_L \text{3-SAT}$$
Reduction from **CIRCUIT SAT** to **3-SAT**

Let an arbitrary instance of **CIRCUIT SAT** be given by a Boolean circuit $C$. We construct the following instance $\varphi(C)$ of **SAT** ($\varphi$ is in CNF with some clauses smaller than 3. The transformation into 3-CNF is obvious):

The formula $\varphi(C)$ uses all variables of $C$. Moreover, for each gate $g$ of $C$, $\varphi(C)$ has a new variable $g$ and the following clauses.

1. If $g$ is a variable gate $x$: $(g \lor \neg x), (\neg g \lor x)$.  
   \[g \leftrightarrow x\]

2. If $g$ is a **true** (resp. **false**) gate: $g$ (resp. $\neg g$).

3. If $g$ is a NOT gate with a predecessor $h$:  
   $(\neg g \lor \neg h), (g \lor h)$.  
   \[g \leftrightarrow \neg h\]

4. If $g$ is an AND gate with predecessors $h, h'$:  
   $(\neg g \lor h), (\neg g \lor h'), (g \lor \neg h \lor \neg h')$.  
   \[g \leftrightarrow (h \land h')\]

5. If $g$ is an OR gate with predecessors $h, h'$:  
   $(\neg g \lor h \lor h'), (g \lor \neg h'), (g \lor \neg h)$.  
   \[g \leftrightarrow (h \lor h')\]

6. If $g$ is also the output gate: $g$.  

---

Reinhard Pichler

19 April, 2016
**Not-all-equal SAT (NAESAT)**

**INSTANCE:** Boolean formula $\varphi$ in 3-CNF

**QUESTION:** Does there exist a truth assignment $T$ appropriate to $\varphi$, such that the 3 literals in each clause do not have the same truth value?

**Remark.** Clearly $\text{NAESAT} \subset \text{3-SAT}$.

**Theorem**

$\text{NAESAT}$ is NP-complete.
NAESAT

Proof of NP-Hardness

Recall the Boolean formula $\varphi(C)$ resulting from the reduction of CIRCUIT SAT to 3-SAT. For all one- and two-literal clauses in the resulting CNF-formula $\varphi(C)$, we add the same literal $z$ (possibly twice) to make them 3-literal clauses.

The resulting formula $\varphi_z(C)$ fulfills the following equivalence:

$$\varphi_z(C) \in \text{NAESAT} \iff C \in \text{CIRCUIT SAT}.$$  

“⇒” If a truth assignment $T$ satisfies $\varphi_z(C)$ in the sense of NAESAT, so does the complementary truth assignment $\overline{T}$.

Thus, $z$ is false in either $T$ or $\overline{T}$ which implies that $\varphi(C)$ is satisfied by either $T$ or $\overline{T}$. Thus $C$ is satisfiable.
Proof of NP-Hardness (continued)

“⇐” If $C$ is satisfiable, then there is a truth assignment $T$ satisfying $\varphi(C)$. Let us then extend $T$ for $\varphi_z(C)$ by assigning $T(z) = \text{false}$.

By assumption, $T$ is a satisfying truth assignment of $\varphi(C)$ and, therefore, also of $\varphi_z(C)$. Hence, in no clause of $\varphi_z(C)$ all literals are $\text{false}$.

It remains to show that in no clause of $\varphi_z(C)$ all literals are $\text{true}$:

(i) Clauses for true/false/NOT/variable gates contain $z$ that is false.

(ii) For AND gates the clauses are: $(\neg g \lor h \lor z)$, $(\neg g \lor h' \lor z)$, $(g \lor \neg h \lor \neg h')$ where in the first two $z$ is $\text{false}$, and in the third all three cannot be $\text{true}$ as then the first two clauses would be $\text{false}$.

(iii) For OR gates the clauses are: $(\neg g \lor h \lor h')$, $(g \lor \neg h' \lor z)$, $(g \lor \neg h \lor z)$ where in the last two $z$ is $\text{false}$, and in the first all three cannot be $\text{true}$ as then the last two clauses would be $\text{false}$.
1-IN-3-SAT

INSTANCE: Boolean formula $\varphi$ in 3-CNF

QUESTION: Does there exist a truth assignment $T$ appropriate to $\varphi$, such that in each clause, exactly one literal is true in $T$?

MONOTONE 1-IN-3-SAT

INSTANCE: Boolean formula $\varphi$ in 3-CNF, s.t. the clauses in $\varphi$ contain only unnegated atoms.

QUESTION: Does there exist a truth assignment $T$ appropriate to $\varphi$, such that in each clause, exactly one literal is true in $T$?

Theorem

Both 1-IN-3-SAT and MONOTONE 1-IN-3-SAT are NP-complete.
Remarks

- Clearly $\textbf{1-IN-3-SAT} \subset \textbf{NAESAT} \subset \textbf{3-SAT}$. The instances of these 3 problems are the same, namely 3-CNF formulae. However, the positive instances of $\textbf{1-IN-3-SAT}$ are a proper subset of $\textbf{NAESAT}$, which in turn are a proper subset of the positive instances of $\textbf{3-SAT}$.

- Note that the NP-completeness of any of these 3 problems does not immediatetely imply the NP-completeness of any of the other problems, since it is a priori not clear if further constraining the positive instances makes things easier or harder.

- $\textbf{MONOTONE 1-IN-3-SAT}$ is a special case of $\textbf{1-IN-3-SAT}$, i.e., the instances of the former are a proper subset of the latter while the question remains the same. The NP-hardness of the special case immediately implies the NP-hardness of the general case.
Proof of the NP-hardness of 1-IN-3-SAT

We prove the NP-hardness by a reduction from 4-SAT: Let $\varphi$ be an arbitrary instance of 4-SAT, i.e., $\varphi$ is in 4-CNF. We construct an instance $\psi$ of 1-IN-3-SAT as follows:

For every clause $l_1 \lor l_2 \lor l_3 \lor l_4$ in $\varphi$, let $a_1, a_2, a_3, a_4, b_1, b_2, c_1, c_2, d$ be 9 fresh propositional variables. Then $\psi$ contains the following 7 clauses:

(1) $l_1 \lor a_1 \lor b_1$
(2) $l_2 \lor a_2 \lor b_1$
(3) $a_1 \lor a_2 \lor c_1$
(4) $l_3 \lor a_3 \lor b_2$
(5) $l_4 \lor a_4 \lor b_2$
(6) $a_3 \lor a_4 \lor c_2$
(7) $b_1 \lor b_2 \lor d$

Idea. Suppose that in a truth assignment $T$ of $\varphi$ all literals in the clause $l_1 \lor \cdots \lor l_4$ are false:

By (1) – (3): If $l_1$ and $l_2$ are false, then $b_1$ must be true.

By (4) – (6): If $l_3$ and $l_4$ are false, then $b_2$ must be true.

However, by (7), it is not allowed that both $b_1$ and $b_2$ are true.
Proof of the NP-hardness of **MONOTONE 1-IN-3-SAT**

We show how an arbitrary instance \( \varphi \) of **1-IN-3-SAT** can be transformed into an equivalent instance \( \psi \) of **MONOTONE 1-IN-3-SAT**:

Let \( X = \{x_1, \ldots, x_n\} \) be the variables in \( \varphi \). Then the variables in \( \psi \) are \( X \cup \{x'_i \mid 1 \leq i \leq n\} \cup \{a, b, c\} \). In \( \varphi \), we replace every negative literal of the form \( \neg x_i \) (for some \( i \)) by the unnegated atom \( x'_i \).

Moreover, for every \( i \in \{1, \ldots, n\} \), we add the following 3 clauses:

1. \( x_i \lor x'_i \lor a \)
2. \( x_i \lor x'_i \lor b \)
3. \( a \lor b \lor c \)

**Idea.** These three clauses guarantee that in a legal 1-in-3 assignment of \( \psi \), the variables \( x_i \) and \( x'_i \) have complementary truth values. Hence, \( x'_i \) indeed encodes \( \neg x_i \).
HITTING SET

INSTANCE: Set \( T = \{t_1, \ldots, t_p\} \), family \((V_i)_{1 \leq i \leq n}\) of subsets of \( T \), i.e.:
for all \( i \in \{1, \ldots, n\} \), \( V_i \subseteq T \).

QUESTION: Does there exist a set \( W \subseteq T \), s.t. \(|W \cap V_i| = 1\) for all \( i \in \{1, \ldots, n\}\)? (A set \( W \) with this property is called a “hitting set”).

Corollary

HITTING SET \( \text{is NP-complete.} \)

Proof of the NP-hardness

By reduction from MONOTONE 1-IN-3-SAT: Let an instance of MONOTONE 1-IN-3-SAT be given by the 3-CNF formula \( \varphi \) over the variables \( X \). We define the following instance of HITTING SET:

\( T = X \). Moreover, suppose that \( \varphi \) contains \( n \) clauses. Then there are \( n \) sets \((V_i)_{1 \leq i \leq n}\). If the \( i \)-th clause in \( \varphi \) is \( l_1 \lor l_2 \lor l_3 \), then \( V_i = \{l_1, l_2, l_3\} \).
Some Graph Problems

We have already proved the NP-completeness of the following graph problems:

- **INDEPENDENT SET**
- **CLIQUE**
- **VERTEX COVER**

We shall now show the following results:

- **3-COLORABILITY** is NP-complete.
- **HAMILTON-PATH** $\leq_L$ **HAMILTON-CYCLE** $\leq_L$ **TSP(D)**
**INDEPENDENT SET**

INSTANCE: Undirected graph $G = (V, E)$ and integer $K$.

QUESTION: Does there exist an independent set $I$ of size $\geq K$?

i.e., $I \subseteq V$, s.t. for all $i, j \in I$ with $i \neq j$, $[i, j] \not\in E$.

---

**CLIQUE**

INSTANCE: Undirected graph $G = (V, E)$ and integer $K$.

QUESTION: Does there exist a clique $C$ of size $\geq K$?

i.e., $C \subseteq V$, s.t. for all $i, j \in C$ with $i \neq j$, $[i, j] \in E$.

---

**VERTEX COVER**

INSTANCE: Undirected graph $G = (V, E)$ and integer $K$.

QUESTION: Does there exist a vertex cover $N$ of size $\leq K$?

i.e., $N \subseteq V$, s.t. for all $[i, j] \in E$, either $i \in N$ or $j \in N$. 
### Decision Problems

#### 3-COLORABILITY

**INSTANCE:** Undirected graph $G = (V, E)$

**QUESTION:** Does $G$ have a 3-coloring? i.e., an assignment of one of 3 colors to each of the vertices in $V$ such that any two vertices $i, j$ connected by an edge $[i, j] \in E$ do not have the same color?

#### k-COLORABILITY (for fixed value $k$)

**INSTANCE:** Undirected graph $G = (V, E)$

**QUESTION:** Does $G$ have a $k$-coloring? i.e., an assignment of one of $k$ colors to each of the vertices in $V$ such that any two vertices $i, j$ connected by an edge $[i, j] \in E$ do not have the same color?
Theorem

The k-COLORABILITY problem is NP-complete for any fixed $k \geq 3$. The 2-COLORABILITY problem is in P.

Proof

NP-Membership of k-COLORABILITY:
1. Guess an assignment $f : V \rightarrow \{1, \ldots, k\}$
2. Check for every edge $[i, j] \in E$ that $f(i) \neq f(j)$.

P-Membership of 2-COLORABILITY: (w.l.o.g., $G$ is connected)
1. Start by assigning an arbitrary color to an arbitrary vertex $v \in V$.
2. Suppose that the vertices in $S \subset V$ have already been assigned a color. Choose $x \in S$ and assign to all vertices adjacent to $x$ the opposite color. $G$ is 2-colorable iff step 2 never leads to a contradiction.
NP-Hardness Proof of 3-COLORABILITY

By reduction from NAESAT: Let an arbitrary instance of NAESAT be given by a Boolean formula $\varphi = c_1 \land \ldots \land c_m$ in 3-CNF with variables $x_1, \ldots, x_n$. We construct the following graph $G(\varphi)$:

Let $V = \{a\} \cup \{x_i, \neg x_i \mid 1 \leq i \leq n\} \cup \{l_{i_1}, l_{i_2}, l_{i_3} \mid 1 \leq i \leq m\}$, i.e. $|V| = 1 + 2n + 3m$.

For each variable $x_i$ in $\varphi$, we introduce a triangle $[a, x_i, \neg x_i]$, i.e. all these triangles share the node $a$.

For each clause $c_i$ in $\varphi$, we introduce a triangle $[l_{i_1}, l_{i_2}, l_{i_3}]$. Moreover, each of these vertices $l_{ij}$ is further connected to the node corresponding to this literal, i.e.: if the $j$-th literal in $c_i$ is of the form $x_\alpha$ (resp. $\neg x_\alpha$) then we introduce an edge between $l_{ij}$ and $x_\alpha$ (resp. $\neg x_\alpha$)
Example

The 3-CNF formula \( \varphi = (x_1 \lor \neg x_2 \lor x_3) \land (x_2 \lor x_3 \lor \neg x_4) \) is reduced to the following graph:
Example

The 3-CNF formula $\varphi = (x_1 \lor \neg x_2 \lor x_3) \land (x_2 \lor x_3 \lor \neg x_4)$ is reduced to the following graph:

Let red = \text{false} and green = \text{true}. The above 3-coloring corresponds to $T(x_1) = T(\neg x_2) = T(\neg x_3) = T(\neg x_4) = \text{true}$. 
Correctness of the Problem Reduction

Proof (continued)

“⇐” Suppose that $G$ has a 3-coloring with colors $\{0, 1, 2\}$. W.l.o.g., the node $a$ has the color 2. This induces a truth assignment $T$ via the colors of the nodes $x_i$: if the color is 1, then $T(x_i) = \text{true}$ else $T(x_i) = \text{false}$. We claim that $T$ is a legal \textbf{NAESAT}-assignment. Indeed, if in some clause, all literals had the value \text{false} (resp. \text{true}), then we could not use the color 0 (resp. 1) for coloring the triangle $[l_{i1}, l_{i2}, l_{i3}]$, a contradiction.

“⇒” Suppose that there exists an \textbf{NAESAT}-assignment $T$ of $\varphi$. Then we can extract a 3-coloring for $G$ from $T$ as follows:

(i) Node $a$ is colored with color 2.
(ii) If $T(x_i) = \text{true}$, then color $x_i$ with 1 and $\neg x_i$ with 0 else vice versa.
(iii) From each $[l_{i1}, l_{i2}, l_{i3}]$, color two literals having opposite truth values with 0 (\text{true}) and 1 (\text{false}). Color the third with 2.
### HAMILTON-PATH

**INSTANCE:** (directed or undirected) graph $G = (V, E)$

**QUESTION:** Does $G$ have a *Hamilton path*?

* i.e., a path visiting all vertices of $G$ exactly once.

### HAMILTON-CYCLE

**INSTANCE:** (directed or undirected) graph $G = (V, E)$

**QUESTION:** Does $G$ have a *Hamilton cycle*?

* i.e., a cycle visiting all vertices of $G$ exactly once.

### TSP(D)

**INSTANCE:** $n$ cities $1, \ldots, n$ and a nonnegative integer distance $d_{ij}$ between any two cities $i$ and $j$ (such that $d_{ij} = d_{ji}$), and an integer $B$.

**QUESTION:** Is there a tour through all cities of length at most $B$?

* i.e., a permutation $\pi$ s.t. $\sum_{i=1}^{n} d_{\pi(i)\pi(i+1)} \leq B$ with $\pi(n + 1) = \pi(1)$. 
Complexity

Theorem

HAMILTON-PATH, HAMILTON-CYCLE, and TSP(D) are NP-complete.

Proof

We shall show the following chain of reductions:

\[ \text{HAMILTON-PATH} \leq_L \text{HAMILTON-CYCLE} \leq_L \text{TSP(D)} \]

It suffices to show NP-membership for the hardest problem:
1. Guess a tour \( \pi \) through the \( n \) cities.
2. Check that \( \sum_{i=1}^{n} d_{\pi(i)\pi(i+1)} \leq B \) with \( \pi(n+1) = \pi(1) \).

Likewise, it suffices to prove the NP-hardness of the easiest problem. The NP-hardness of HAMILTON-PATH (by a reduction from 3-SAT) is quite involved and is therefore omitted here (see Papadimitriou’s book).
HAMILTON-PATH \textbf{vs.} HAMILTON-CYCLE

\begin{center}
\begin{tabular}{|c|}
\hline
HAMILTON-PATH \leq_L HAMILTON-CYCLE \\
\hline
\end{tabular}
\end{center}

(We only consider undirected graphs). Let an arbitrary instance of HAMILTON-PATH be given by the graph $G = (V, E)$. We construct an equivalent instance $G' = (V', E')$ of HAMILTON-CYCLE as follows:

Let $V' := V \cup \{z\}$ for some new vertex $z$ and $E' := E \cup \{[v, z] \mid v \in V\}$. 

$G$ has a Hamilton path $\iff G'$ has a Hamilton cycle

$\Rightarrow$ Suppose that $G$ has a Hamilton path $\pi$ starting at vertex $a$ and ending at $b$. Then $\pi \cup \{z\}$ is clearly a Hamilton cycle in $G'$.

$\Leftarrow$ Let $C$ be a Hamilton cycle in $G'$. In particular, $C$ goes through $z$. Let $a$ and $b$ be the two neighboring nodes of $z$ in this cycle. Then $C \setminus \{z\}$ is a Hamilton path (starting at vertex $a$ and ending at $b$) in $G$. 

Reinhard Pichler

19 April, 2016
**HAMILTON-CYCLE vs. TSP(D)**

**HAMILTON-CYCLE \( \leq_L \) TSP(D)**

Let an arbitrary instance of **HAMILTON-CYCLE** be given by the graph \( G = (V, E) \). We construct an equivalent instance of **TSP(D)** as follows:

Let \( V = \{1, \ldots, n\} \). Then our instance of **TSP(D)** has \( n \) cities. Moreover, for any two cities \( i \neq j \), the distance is defined as

\[
    d_{ij} = \begin{cases} 
    1 & \text{if } [i, j] \in E \\
    2 & \text{otherwise}
    \end{cases}
\]

Finally, we set \( B = n \).

Clearly, there is no tour through all cities of length \( < B = n \).

Moreover, the Hamilton cycles in \( G \) are precisely the tours of length \( B \).

Hence, \( G \) has a Hamilton cycle \( \iff \) there exists a tour of length \( \leq B \).
Summary of Reductions

- **SAT**
  - 4-SAT
  - 1-in-3-SAT
    - MON 1-in-3-SAT
    - HITTING SET
  - 3-SAT
    - INDEPENDENT SET
    - VERTEX COVER
    - CLIQUE
    - HAM.-PATH
      - HAM.-CYCLE
        - TSP(D)
  - CIRCUIT-SAT
    - NAESAT
      - 3-COL
Learning Objectives

- The concept of NP-completeness and its characterizations in terms of succinct certificates.
- You should now be familiar with the intuition of NP-completeness (and recognize NP-complete problems)
- Basic techniques to prove problems NP-complete
- A basic repertoire of NP-complete problems (in particular, versions of SAT and some graph problems) to be used in further NP-completeness proofs.
- Reductions, reductions, reductions, ...