Aalto University
School of Science

CS-E4530 Computational Complexity Theory

Lecture 8: More NP-Complete Problems

Aalto University
School of Science
Department of Computer Science

Spring 2019

Agenda

- More variants of satisfiability
- More graph-theoretic problems
- Sets and numbers

1. More Variants of Satisfiability

- 2SAT
- Not-All-Equal SAT (NAESAT)

2SAT

- 2SAT can be decided in polynomial time by an algorithm determining reachability in a graph associated with a given 2CNF formula ϕ.

Definition

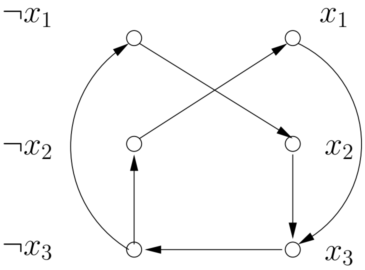
Let ϕ be an instance of 2SAT.
Define a graph $G(\phi)$ as follows:
– The vertices of $G(\phi)$ correspond to the variables of ϕ and their negations.
– For every clause $\alpha \vee \beta$ in ϕ, there are arcs $(\overline{\alpha}, \beta)$ and $(\overline{\beta}, \alpha)$ in $G(\phi)$.

Theorem

Let ϕ be an instance of 2SAT.
Then ϕ is unsatisfiable iff there is a variable x such that there are paths from x to $\neg x$ and from $\neg x$ to x in $G(\phi)$.

Example

- Consider the formula
  $\phi = (x_1 \vee x_2) \wedge (\neg x_1 \vee x_3) \wedge (\neg x_2 \vee x_3) \wedge (\neg x_3 \vee \neg x_3)$
- The graph $G(\phi)$:



- ϕ is unsatisfiable as there is a path from x_3 to $\neg x_3$ and from $\neg x_3$ to x_3 in $G(\phi)$.

2SAT is in P

Corollary

2SAT is in $\mathbf{P}$.

Proof.

The reachability condition of the preceding theorem can be tested by standard graph algorithms (e.g. depth-first-search) in polynomial time. □

Not-All-Equal SAT (NAESAT)

In the NAESAT problem, a given 3CNF formula ϕ is considered satisfied if there is a truth assignment so that in each clause of ϕ, the three literals do not have the same truth value.

Theorem

NAESAT is NP-complete.

Proof. Reduction from 3SAT. (Exercise.)

2. More Graph-Theoretic Problems

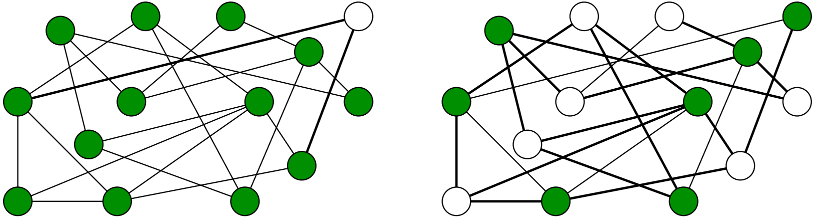
- MIN CUT and MAX CUT
- MAX BISECTION and BISECTION WIDTH
- HAMILTON PATH and TSP

MIN CUT and MAX CUT

- A cut in an undirected graph $G = (V, E)$ is a partition of the vertices into two nonempty sets S and $V - S$.
- The size of a cut is the number of edges between S and $V - S$.

Example

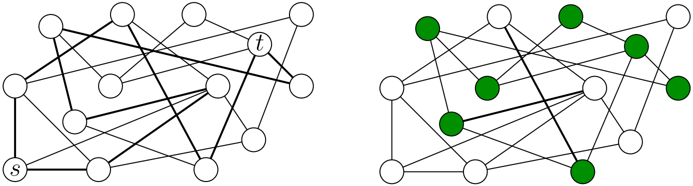
A graph and two cuts (of sizes 2 and 17, resp.):



- The problem of finding a cut with the smallest size is in $\mathbf{P}$:
  (i) The size of the smallest cut that separates two given vertices s and t equals the maximum flow from s to t. ("Max-Flow/Min-Cut Thm".)
  (ii) Minimum cut: find the maximum flow between a fixed s and all other vertices and choose the smallest value found.

Example

A maximum flow and cut of size 2:



- However, the problem of deciding whether there is a cut of a size at least K (MAX CUT) is much harder:

Theorem

MAX CUT is NP-complete.

Reduction from NAESAT to MAX CUT

The NP-completeness of MAX CUT is shown for graphs with multiple edges between vertices by a reduction from NAESAT.

- For a conjunction of clauses $\phi = C_1 \wedge \ldots \wedge C_m$, we construct a graph $G = (V,E)$ so that

  G has a cut of size $5m$ iff ϕ is satisfied in the sense of NAESAT.

- The vertices of G are $x_1, \ldots, x_n, \neg x_1, \ldots, \neg x_n$ where $x_1, \ldots, x_n$ are the variables in ϕ.
- The edges in G include a triangle $[\alpha, \beta, \gamma]$ for each clause $\alpha \vee \beta \vee \gamma$ and n_i copies of the edge $\{x_i, \neg x_i\}$ where n_i is the number of occurrences of x_i or $\neg x_i$ in the clauses.
- Now a cut $(S, V-S)$ of size $5m$ in G corresponds to a truth assignment satisfying ϕ in the sense of NAESAT.

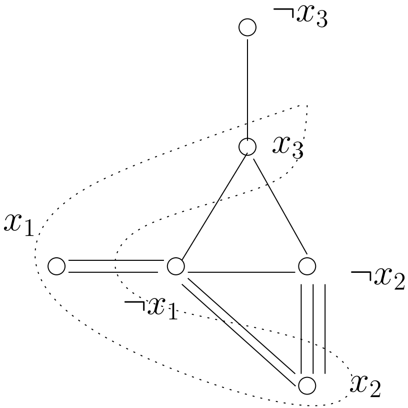
Example. Consider the conjunction of clauses ϕ:

$$(\neg x_1 \vee x_2 \vee x_2) \wedge (\neg x_1 \vee \neg x_2 \vee x_3)$$

which is satisfied in the sense of NAESAT iff the graph G on the right obtained as the result of the reduction has a cut of size 5*2=10.
For instance,

$$(\{x_1, x_2, x_3\}, \{\neg x_1, \neg x_2, \neg x_3\})$$

is a cut of size 10 and it corresponds to a truth assignment $T(x_1) = T(x_2) = T(x_3) = \textbf{true}$ satisfying ϕ in the sense of NAESAT.



Correctness of the reduction

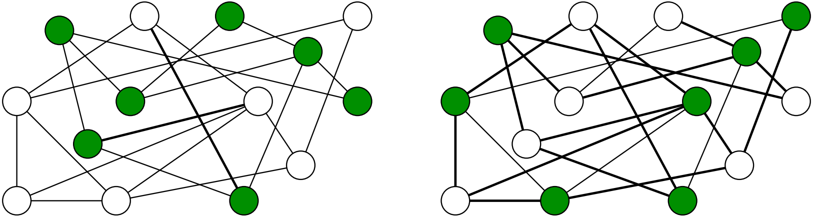
- It is easy to see that a satisfying truth assignment (in the sense of NAESAT) gives rise to a cut of size $5m$.
- Conversely, suppose there is a cut $(S, V-S)$ of size $5m$ or more.
- All variables can be assumed separate from their negations:
  If both $x_i, \neg x_i$ are on the same side, they contribute at most $2n_i$ edges to the cut (where n_i is the number of occurrences of x_i or $\neg x_i$ in the clauses).
  Hence, moving the one with fewer neighbours to the other side of the cut does not decrease the size of the cut.
- Let S be the set of true literals and $V-S$ those false.
- The total number of edges in the cut joining opposite literals is $3m$. The remaining $2m$ are coming from triangles meaning that all m triangles are cut, i.e. ϕ is satisfied in the sense of NAESAT. $\square$

Graph problems: MAX BISECTION

- In many applications of graph partitioning, the sizes of S and $V - S$ cannot be arbitrarily small or large.
- MAX BISECTION is the problem of determining whether there is a cut $(S, V - S)$ with size of K or more such that $|S| = |V - S|$.

Example

Bisections with cut sizes of 2 and 17, respectively:



- Is MAX BISECTION easier than MAX CUT?

Lemma

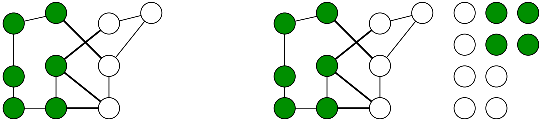
MAX BISECTION is NP-complete.

Proof.

Reducing MAX CUT to MAX BISECTION by modifying input:
Add $|V|$ disconnected new vertices to G. Now every cut of G can be made a bisection by appropriately splitting the new vertices.
Now $G = (V, E)$ has a cut $(S, V - S)$ with size of K or more iff the modified graph has a cut with size of K or more with $|S| = |V - S|$. □

Example

Reducing MAX CUT to MAX BISECTION:



Graph problems: BISECTION WIDTH

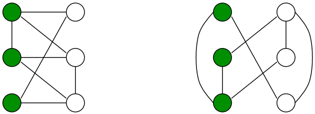
- The respective minimisation problem, i.e. MIN CUT with the bisection requirement, is NP-complete, too.
  (Remember that MIN CUT $\in \mathbf{P}$).
- BISECTION WIDTH: is there a bisection of size K or less?

Theorem

BISECTION WIDTH is NP-complete.

Proof.

A reduction from MAX BISECTION. A graph $G = (V, E)$ where $|V| = 2n$ for some n has a bisection of size K or more iff the complement $\overline{G}$ has a bisection of size $n^2 - K$ or less. □



Graph problems: HAMILTON PATH

Theorem

HAMILTON PATH is NP-complete.

Proof.

- Reduction from 3SAT to HAMILTON PATH: given a formula ϕ in CNF with variables $x_1, \ldots, x_n$ and clauses $C_1, \ldots, C_m$ each with three literals, we construct a graph $R(\phi)$ that has a Hamilton path iff ϕ is satisfiable.
- *Choice gadgets* select a truth assignment for variables x_i.
- *Consistency gadgets* (XOR) enforce that all occurrences of x_i have the same truth value and all occurrences of $\neg x_i$ the opposite.
- *Constraint gadgets* guarantee that all clauses are satisfied.

Gadgets [Papadimitriou, 1994]

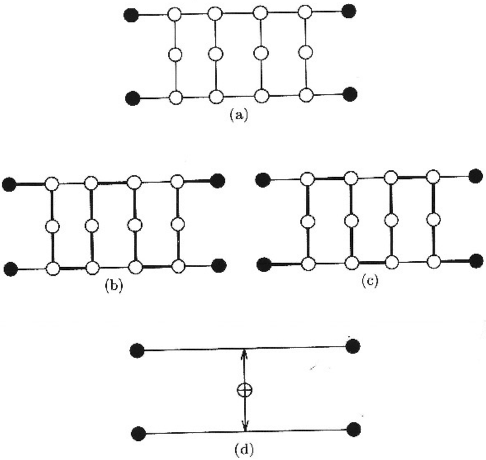


Figure 9-5. The consistency gadget.

Figure 9-4. The choice gadget.

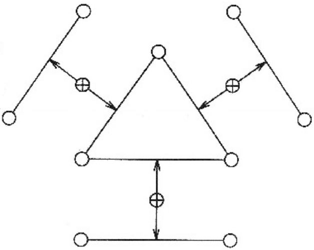


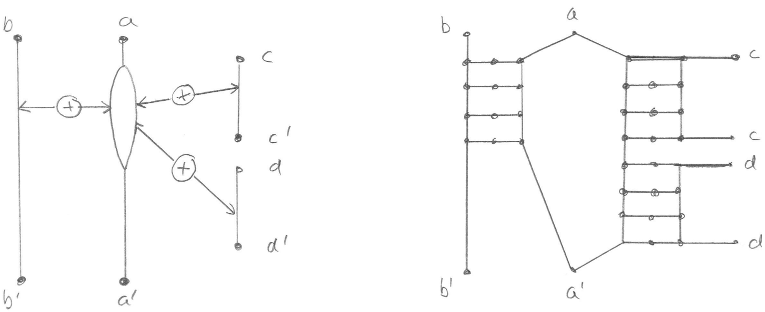
Figure 9-6. The constraint gadget.

Reduction from 3SAT to HAMILTON PATH

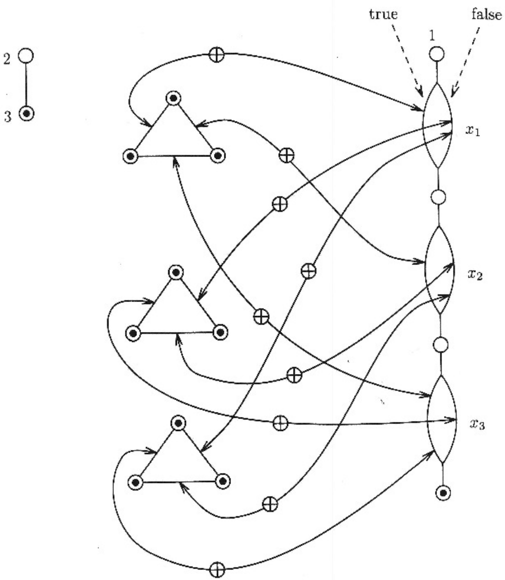
The graph $R(\phi)$ is constructed as follows:

- The *choice gadgets* of variables x_i are connected in series.
- A *constraint gadget* (triangle) for each clause with an edge identified with each literal l in the clause.
  – If l is x_i, then XOR to $\mathbf{true}$ edge of choice gadget of x_i.
  – If it is $\neg x_i$, then XOR to $\mathbf{false}$ edge of choice gadget of x_i.
- All vertices of the triangles, the end vertex of choice gadgets and a new vertex 3 form a clique. Add a vertex 2 connected to 3.

Basic idea: each side of the constraint gadget is traversed by the Hamilton path iff the corresponding literal is $\mathbf{false}$. Hence, at least one literal in any clause is $\mathbf{true}$ since otherwise all sides for its triangle should be traversed which is impossible (implying no Hamilton path).


b
a
c
c'
d
d'
b'
a'
b
a
c
c
d
d
b'
a'

$(x_1 \vee x_2 \vee x_3) \wedge (\neg x_1 \vee \neg x_2 \vee \neg x_3) \wedge (\neg x_1 \vee \neg x_2 \vee x_3)$



[Papadimitriou, 1994]

Correctness of the reduction

- If ϕ is satisfiable, there is a Hamilton path:
  From a satisfying truth assignment, we construct a Hamilton path by starting at 1, traversing choice gadgets according to the truth assignment, the rest is a big clique for which a trivial path can be found leading to 3 and then to 2.
- If there is a Hamilton path, ϕ is satisfiable:
  The path starts at 1, makes a truth assignment, traverses the triangles in some order and ends up in 2. The truth assignment satisfies ϕ as there is no triangle where all sides are traversed, i.e., where all literals are **false**. □

Travelling salesperson (TSP) revisited

Corollary

TSP(D) is NP-complete.

Proof: A reduction from HAMILTON PATH to TSP(D). Given a graph G with n vertices, construct a distance matrix d_{ij} and a budget B so that there is a tour of length at most B iff G has a Hamilton path.

- There are n cities and the distance $d_{ij} = 1$ if there is $\{i,j\} \in G$ and $d_{ij} = 2$ otherwise. The budget $B = n+1$.
- If there is a tour of length $n+1$ or less, then there is at most one pair $(\pi(i), \pi(i+1))$ in it with cost 2, i.e., a pair for which $\{\pi(i), \pi(i+1)\}$ is not an edge. Removing it gives a Hamilton path.
- If G has a Hamilton path, then its cost is $n-1$ and it can be made into a tour with additional cost of at most 2. □

3. Sets and Numbers

- TRIPARTITE MATCHING
- EXACT COVER BY 3-SETS
- KNAPSACK
- Pseudopolynomial algorithms
- Strong NP-completeness
- BIN PACKING

Sets and numbers: TRIPARTITE MATCHING

Definition

TRIPARTITE MATCHING:
INSTANCE: Three sets B (boys), G (girls), and H (houses) each containing n elements, and a ternary relation $T \subseteq B \times G \times H$.
QUESTION: Is there a set of n triples in T no two of which have a component in common?

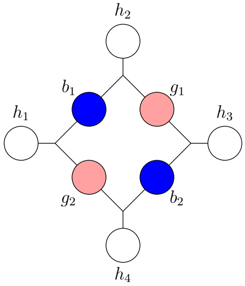
Theorem

TRIPARTITE MATCHING is NP-complete.

Proof. By reduction from 3SAT. Each variable x has a combined choice and consistency gadget, and each clause c a dedicated pair of boy b_c and girl g_c, together with three triples (b_c, g_c, h_l) where h_l ranges over the three houses corresponding to the occurrences of literals in the clause (appearing in the combined gadgets).

The combined gadget for choice and consistency

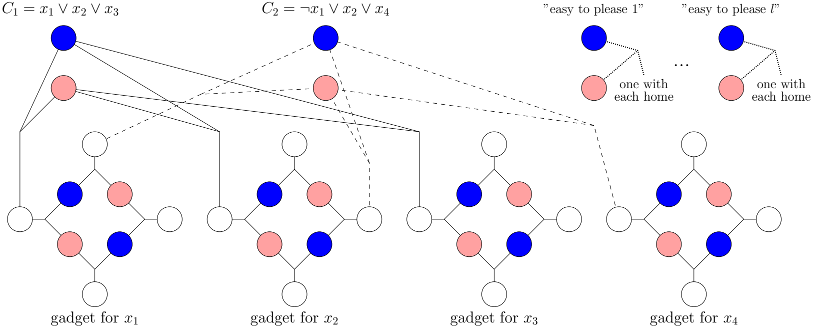
The gadget for a variable x involves k boys, k girls and $2k$ houses forming a "k-circle", where k is either the number of occurrences of x or its negation whichever is larger. (Recall that k can be assumed to equal 2.) The case $k = 2$ is given alongside.



- Occurrences of x in the clauses are connected to the odd houses h_{2i-1} in the variable gadget for x and those of $\neg x$ to the even houses h_{2i}.
- Exactly two kinds of matchings in the variable gadget for x are possible:
  – "$T(x) = \textbf{true}$": each b_i with g_i and h_{2i}.
  – "$T(x) = \textbf{false}$": each b_i with g_{i-1} (g_k if $i = 1$) and h_{2i-1}.

Example

Reducing 3SAT to TRIPARTITE MATCHING:



Correctness of the reduction

- Note that a "$T(x) = \textbf{true}$" matching in the variable gadget for x leaves the odd houses unoccupied, and a "$T(x) = \textbf{false}$" matching respectively the even houses.
- For a clause c, the dedicated b_c and g_c can be matched to a house h in a variable gadget for x that is left unoccupied when x is assigned a truth values satisfying c.
- One more detail needs to be settled: there are now more houses H than boys B and girls G (but $|B| = |G|$).
- Solution: add $l = |H| - |B|$ new boys and l new girls. The ith new girl participates in $|H|$ triples containing the ith new boy and each house.
- Now a tripartite matching exists iff the set of clauses is satisfiable.

Sets and numbers: EXACT COVER BY 3-SETS

Definition

EXACT COVER BY 3-SETS:
INSTANCE: A family $F = \{S_1, \ldots, S_n\}$ of subsets of a finite set U such that $|U| = 3m$ for some integer m and $|S_i| = 3$ for all i.
QUESTION: Is there a subfamily of m sets in F that are disjoint and have U as their union?

Corollary

EXACT COVER BY 3-SETS is NP-complete.

sketch.

TRIPARTITE MATCHING can be reduced to EXACT COVER BY 3-SETS by noticing that it is a special case where U is partitioned in three sets B, G, H with $|B| = |G| = |H|$ and
$F = \{\{b, g, h\} \mid (b, g, h) \in T\}$. □

Example

TRIPARTITE MATCHING:
$B = \{b_1, ..., b_n\}, G = \{g_1, ..., g_n\}$,
$H = \{h_1, ..., h_n\}$,
$T = \{(b_1, g_2, h_1), (b_1, g_2, h_2), ...\}$

EXACT COVER BY 3-SETS:
$U = \{b_1, ..., b_n, g_1, ..., g_n, h_1, ..., h_n\}$
$F = \{\{b_1, g_2, h_1\}, \{b_1, g_2, h_2\}, ...\}$

Sets and numbers: KNAPSACK

Definition

KNAPSACK:
INSTANCE: A set of n items with each item i having a value v_i and a weight w_i (both positive integers) and integers W and K.
QUESTION: Is there a subset S of the items such that $\Sigma_{i \in S} w_i \leq W$ but $\Sigma_{i \in S} v_i \geq K$?

Theorem

KNAPSACK is NP-complete.

Proof. We show that a simple special case of KNAPSACK is NP-complete where $v_i = w_i$ for all i and $W = K$:
INSTANCE: A set of integers $w_1, \ldots, w_n$ and an integer K.
QUESTION: Is there a subset S of the integers with $\Sigma_{i \in S} w_i = K$?

Reduction from EXACT COVER BY 3-SETS

The reduction is based on the set $U = \{1, 2, \ldots, 3m\}$ and the sets $S_1, \ldots, S_n$ given as bit vectors $\{0,1\}^{3m}$ and $K = 2^{3m} - 1$. Then the task is to find a subset of bit vectors that sum to K.

$$
\begin{array}{ccccc}
\rightarrow & 0 & 1 & \ldots & 0 & 0 \\
 & 1 & 0 & \ldots & 0 & 0 \\
 & \vdots & & & & \\
\rightarrow & 0 & 0 & \ldots & 1 & 1 \\
\hline
 & 1 & 1 & \ldots & 1 & 1
\end{array}
$$

- This does not quite work because of the carry bit, but the problem can be circumvented by using $n+1$ as the base rather than 2.
- Now each S_i corresponds to $w_i = \Sigma_{j \in S_i} (n+1)^{3m-j}$.
- Then a set of these integers w_i adds up to $K = \sum_{j=0}^{3m-1} (n+1)^j$ iff there is an exact cover among $\{S_1, S_2, \ldots, S_n\}$. $\square$

Example

Reducing EXACT COVER BY 3-SETS to KNAPSACK

EXACT COVER BY 3-SETS:

$U = \{e_1, \ldots, e_6\}$

$F = \{S_1 = \{e_1, e_4, e_6\}, S_2 = \{e_1, e_3, e_6\}, S_3 = \{e_2, e_3, e_5\}\}$

reduces to

KNAPSACK:

Integers

$$w_1 = 1 \cdot 4^{6-6} + 0 \cdot 4^{6-5} + 1 \cdot 4^{6-4} + 0 \cdot 4^{6-3} + 0 \cdot 4^{6-2} + 1 \cdot 4^{6-1} = 1041$$
$$w_2 = 1 \cdot 4^{6-6} + 0 \cdot 4^{6-5} + 0 \cdot 4^{6-4} + 1 \cdot 4^{6-3} + 0 \cdot 4^{6-2} + 1 \cdot 4^{6-1} = 1089$$
$$w_3 = 0 \cdot 4^{6-6} + 1 \cdot 4^{6-5} + 0 \cdot 4^{6-4} + 1 \cdot 4^{6-3} + 1 \cdot 4^{6-2} + 0 \cdot 4^{6-1} = 324$$

$K = 4^0 + 4^1 + 4^2 + 4^3 + 4^4 + 4^5 = 1365$

Sets and numbers: Pseudopolynomial algorithms

Proposition

Any instance of KNAPSACK can be solved in $O(nW)$ time where n is the number of items and W is the weight limit.

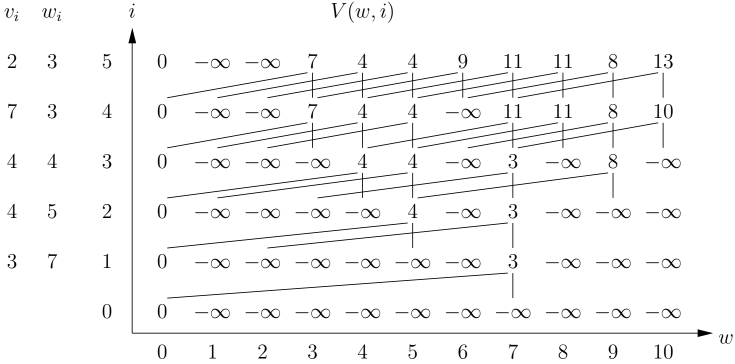
Proof.

- Define $V(w,i)$: the largest value attainable be selecting some among the first i items so that their total weight is exactly w.
- Each $V(w,i)$ with $w = 1,\ldots,W$ and $i = 1,\ldots,n$ can be computed by
$$V(w,i+1) = \max\{V(w,i), v_{i+1} + V(w - w_{i+1}, i)\}$$
where $V(w,i) = -\infty$ if $w \leq 0$, $V(0,i) = 0$ for all i, and $V(w,0) = -\infty$ if $w \geq 1$.
- For each entry this can be done in constant number of steps and there are nW entries. Hence, the algorithm runs in $O(nW)$ time.
- An instance is answered "yes" iff there is an entry $V(w,i) \geq K$.

□

Pseudopolynomial algorithm for KNAPSACK: example

Items $\{(v_1 = 3, w_1 = 7), (v_2 = 4, w_2 = 5), (v_3 = 4, w_3 = 4),$
$(v_4 = 7, w_4 = 3), (v_5 = 2, w_5 = 3)\}$
weight limit $W = 10$, capacity limit $K = 12$



Sets and numbers: Strong NP-completeness

- The preceding algorithm is not polynomial w.r.t. the length of the input (which is $O(n \log W)$) but exponential ($W = 2^{\log W}$).
- An algorithm where the time bound is polynomial in the integers in the input (not their logarithms) is called *pseudopolynomial*.
- A problem is called **strongly NP-complete** if the problem remains NP-complete even if any instance of length n is restricted to contain integers of size (i.e. "value") at most $p(n)$, for a polynomial p.

  ☞ Strongly NP-complete problems cannot have pseudopolynomial algorithms (unless $\mathbf{P} = \mathbf{NP}$).
- SAT, MAX CUT, TSP(D), HAMILTON PATH, $\ldots$ are strongly NP-complete but KNAPSACK is not.

Sets and numbers: BIN PACKING

Definition

BIN PACKING
INSTANCE: N positive integers $a_1, \ldots, a_N$ (items) and integers C (capacity) and B (number of bins).
QUESTION: Is there a partition of the numbers into B subsets such that for each subset S, $\Sigma_{a_i \in S} a_i \leq C$?

- BIN PACKING is strongly NP-complete:
  Even if the integers are restricted to have polynomial values (w.r.t. the length of input), BIN PACKING remains NP-complete.
  For the proof, see the pages 204–205 in Papadimitriou's book.
- Any pseudopolynomial algorithm for BIN PACKING would yield a polynomial algorithm for all problems in NP implying $\mathbf{P} = \mathbf{NP}$.