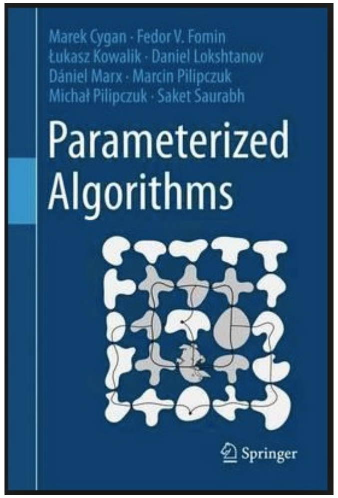
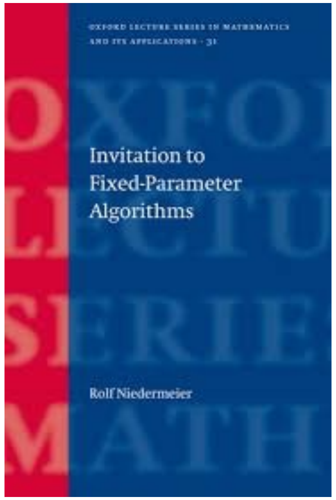
COMPSCI 720: Parameterized and approximation algorithms

Simone Linz

s.linz@auckland.ac.nz



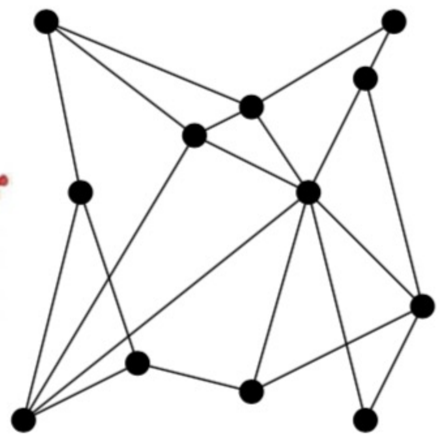


Both books available online from UoA library website.

Fire station problem (Dominating Set)

Every city needs a fire station – or at least one in the neighboring city.

How many fire stations are needed?

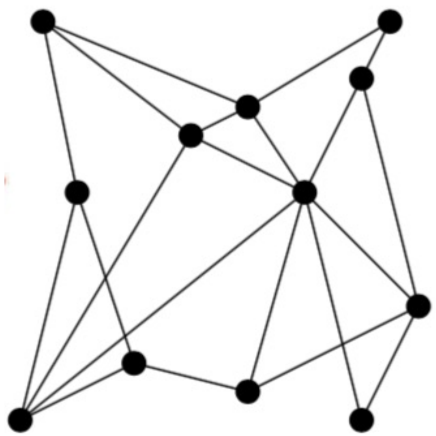


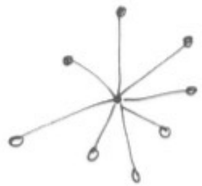
Fire station problem (Dominating Set)

Model the problem as a graph:

Each city is a vertex and two vertices are connected if the two corresponding cities are neighbors of each other.

Example 1.





P_6

K_3 $(=C_3)$

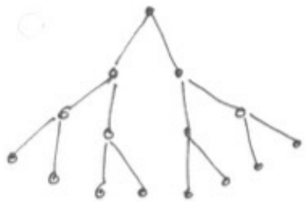
K_4

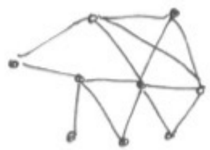
K_5

C_4

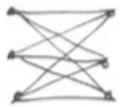
C_5

$G_{2,5}$

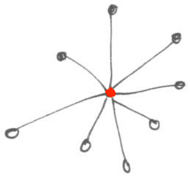




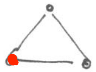
$K_{2,3}$



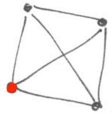
$K_{3,3}$



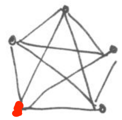
P_6



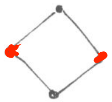
K_3
$(=C_3)$



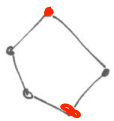
K_4



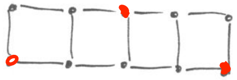
K_5



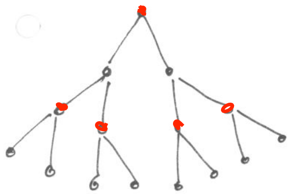
C_4

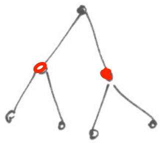


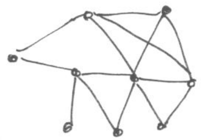
C_5

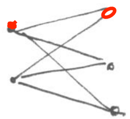


$G_{2,5}$

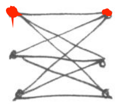








$K_{2,3}$

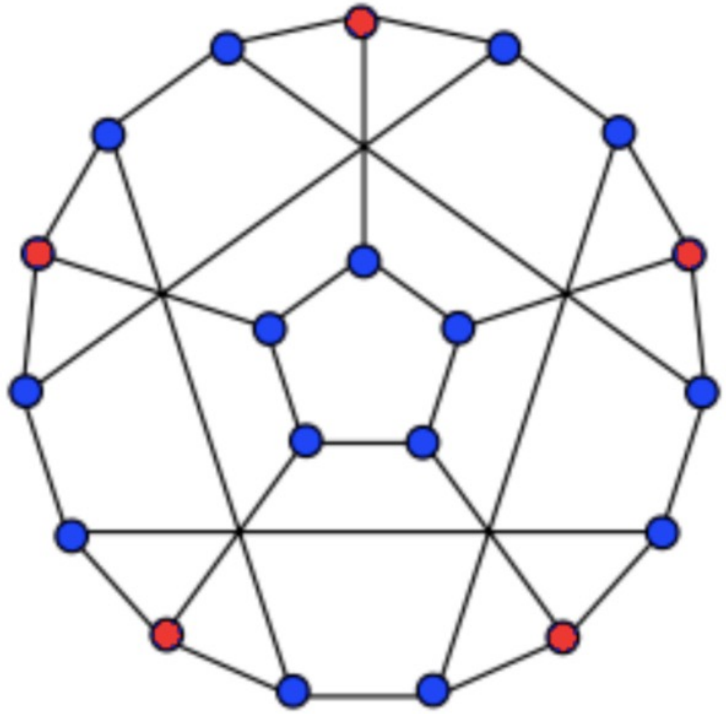


$K_{3,3}$

Dominating Set (DS)

Instance. A graph *G=(V, E)* and a non-negative integer *k*.

Question. Is there a subset *V'* of *V* such that *|V'|≤ k* and each vertex in *V\V'* is adjacent to at least one vertex in *V'*?



Let V' contain the five red vertices. Then each vertex not in V' is adjacent to a red vertex.

Can we do better?

Dominating Set (DS)

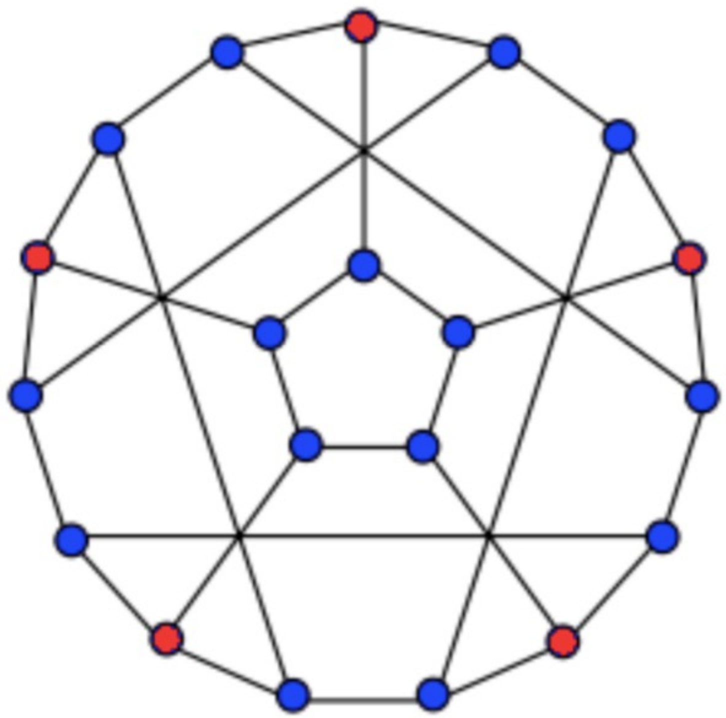
Instance. A graph *G=(V, E)* and a non-negative integer *k*.

Question. Is there a subset *V'* of *V* such that *|V'|≤ k* and each vertex in *V\V'* is adjacent to at least one vertex in *V'*?

Let V' contain the five red vertices. Then each vertex not in V' is adjacent to a red vertex.

Can we do better?

No, V' is a minimum-size dominating set



DS: easy or hard?

Easy – Then there is an algorithm that solves DS in **polynomial** running time.

e.g. $O(n^c)$, where n is the number of vertices (cities) and c is a constant

No such algorithm known for DS!

Hard – Then all known algorithms to solve DS have **exponential** running time.

(Brute force: Check all 2^n subsets of the vertices. No significantly better algorithm known!)

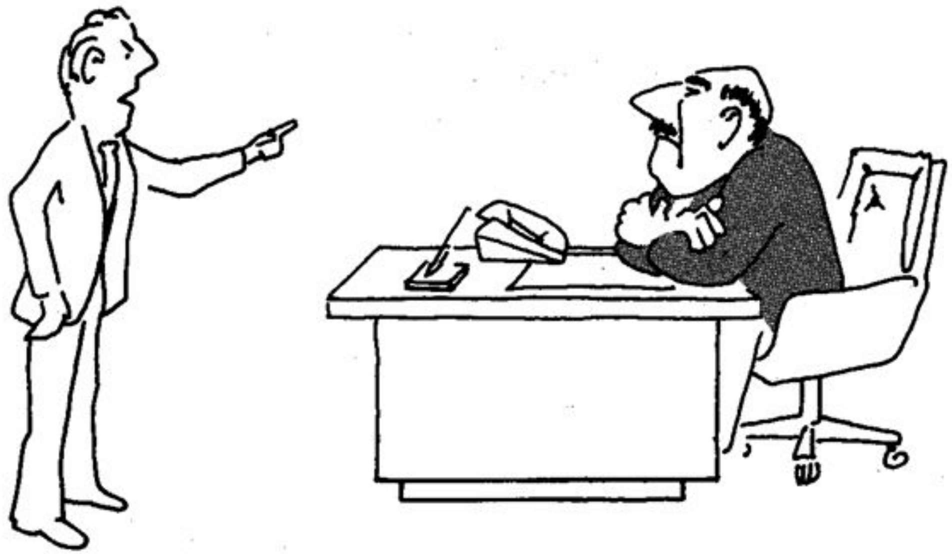
Indeed, DS is NP-complete!

Running times

	Polynomial (e.g. n^2)		Exponential (e.g. 2^n)	
	n^2	time	2^n	time
$n=2$	4	0.04 µs	4	0.04 µs
$n=10$	100	1 µs	1024	10 µs
$n=50$	2500	25 µs	10^{15}	116 days
$n=100$	10000	0.1 ms	10^{30}	3 (10^{14}) years

Dealing with NP-completeness

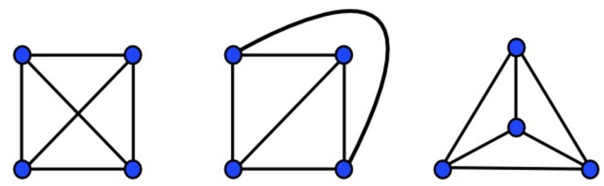
- Never a reason to give up!
- Parameterized algorithms
- Heuristics
- Approximation algorithms (later in course)



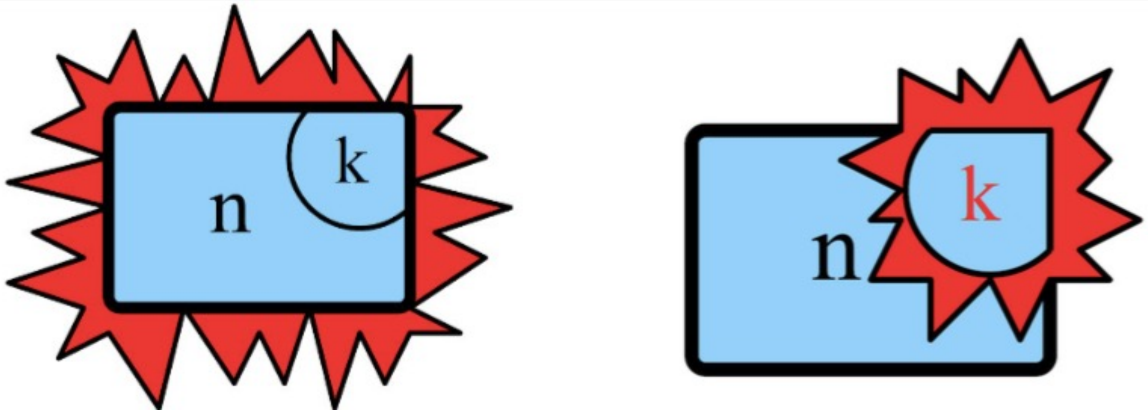
"I can't find an efficient algorithm, because no such algorithm is possible!"

Here: Focus on the design of parameterized algorithms (not so much the complexity theoretic background).

Dominating Set on planar graphs (DS-p)



- NP-complete (no algorithm know that solves the problem in polynomial time).
- BUT: if there exists a *small* dominating set (if only a few fire stations are needed) then the problem can be solved *reasonably quickly*.
- Why? Running time is exponential but the exponential explosion is due to a particular parameter, here *k* (i.e. number of fire stations needed).
- Recognizing if a graph is planar can be done in polynomial time (Kuratowski's Theorem).



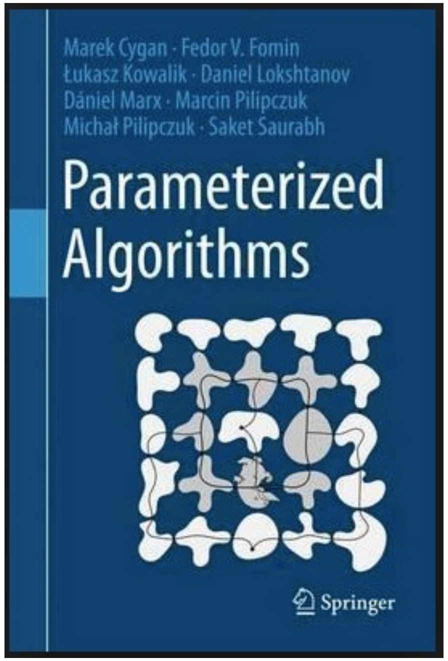
- If the parameter (*k*) is small, the running time is (still) exponential but *often not too bad in practice*.
- Name of the game in parameterized algorithmics: find parameters such that exponential explosion is restricted to these parameters (multidimensional approach, e.g. for DS-p: size of graph and size of solution). Traditional approach is one-dimensional, i.e. running time is measured with respect to input size only.
- Possible parameters: the size of a solution itself (e.g. for DS-p it is the minimum size of a dominating set), tree-width of a graph, diameter of a graph, maximum vertex-degree ….

Definitions

- A parameterized problem is a language $L \subseteq \Sigma^* \times \mathbb{N}$, where Σ is a fixed and finite alphabet. E.g. $(x,k) \in \Sigma^* \times \mathbb{N}$, where k is called the parameter. ($(x,k) \in L$ if and only if (x,k) is a yes-instance)

- A parameterized problem $L \subseteq \Sigma^* \times \mathbb{N}$ is called fixed-parameter tractable (FPT) for parameter k if there exists an algorithm that solves the problem in time $O(f(k)\ |x|^c)$, i.e. one can decide in time $f(k)\ |x|^c$ whether or not $(x,k) \in L$. [Alternative definition uses $f(k) + |x|^c$]

c constant; $f(k)$ computable function <u>only</u> depends on k (e.g. 2^k, 3^k, 2^{2^k})



Chapter 2.

(1) Kernelization

Idea. pre-processing/reducing/shrinking the input data
to the stuff that is difficult to solve

The next three slides are from a presentation by Professor Rod Downey (Victoria University of Wellington).

http://homepages.mcs.vuw.ac.nz/~downey/wa_2011.pdf

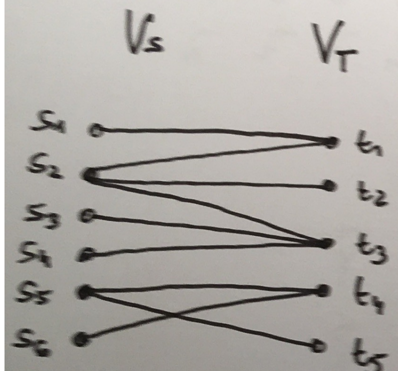
https://royalsociety.org.nz/what-we-do/medals-and-awards/medals-and-awards-news/2018-rutherford-medal-solving-cant-compute-and-is-that-random-sequence-really-random/

Karsten Weihe's Train Problem

- Train Covering by Stations
  Instance: A bipartite graph $G = (V_S \cup V_T, E)$, where the set of vertices V_S represents railway stations and the set of vertices V_T represents trains. E contains an edge $(s, t), s \in V_s, t \in V_T$, iff the train t stops at the station s.
  Problem: Find a minimum set $V' \subseteq V_S$ such that V' covers V_T, that is, for every vertex $t \in V_T$, there is some $s \in V'$ such that $(s, t) \in E$.

WEIHE'S SOLUTION

- REDUCTION RULE TCS1:
  Let $N(t)$ denote the neighbours of t in V_S. If $N(t) \subseteq N(t')$ then remove t' and all adjacent edges of t' from G. If there is a station that covers t, then this station also covers t'.
- REDUCTION RULE TCS2:
  Let $N(s)$ denote the neighbours of s in V_T. If $N(s) \subseteq N(s')$ then remove s and all adjacent edges of s from G. If there is a train covered by s, then this train is also covered by s'.



$V' = \{s_2, s_5\}$

TCS1 (train deletion)

$N(t_1) = \{s_1, s_2\}$

$N(t_2) = \{s_2\}$

$\Rightarrow$ delete t_1

because $N(t_2) \subseteq N(t_1)$

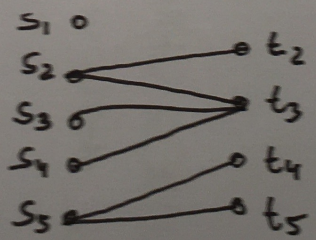
TCS2 (station deletion)

$N(s_5) = \{t_4, t_5\}$

$N(s_6) = \{t_4\}$

$\Rightarrow$ delete s_6

because $N(s_6) \subseteq N(s_5)$



$V' = \{s_2, s_5\}$

Apply TCS1, TCS2 again!

- European train schedule, gave a graph consisting of around $1.6 \cdot 10^5$ vertices and $1.6 \cdot 10^6$ edges.
- Solved in minutes.
- This has also been applied in practice as a subroutine in practical heuristical algorithms.

Idea of kernelization

- in polynomial time, pre-process/reduce/shrink the input data to the stuff that is difficult to solve
- more precisely, reduce the parameterized problem to a kernel whose size depends solely on the parameter
- pre-processing is often used to design heuristics. But we want to do it in a controlled way, so that we get a provable performance guarantee
- want to design reductions such that, at each step, enough information is preserved to obtain an optimal solution

Vertex Cover (VC)

Instance. A graph $G=(V, E)$ and a non-negative integer k.

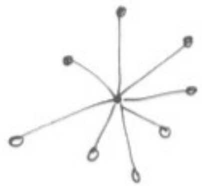
Question. Is there a subset V' of V such that $|V'| \leq k$ and for each edge $\{u,v\}$ in E, at least one of u and v is in V'?

Subtle difference between VC and DS.

VC: Vertices cover edges.

DS: Vertices cover vertices.

Example 2.



P_6

K_3
$(= C_3)$

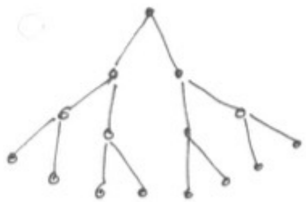
K_4

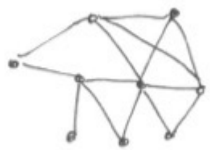
K_5

C_4

C_5

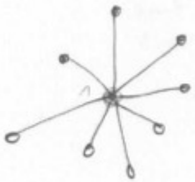
$G_{2,5}$





$K_{2,3}$

$K_{3,3}$



3 P_6

2

K_3 $(=C_3)$

3

K_4

4

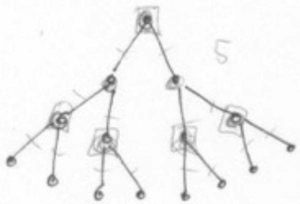
K_5

C_4 2

3

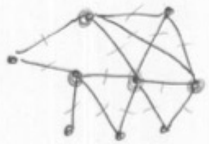
C_5

5 $G_{2,5}$

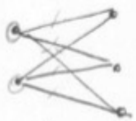


5

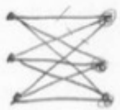
2



4



2 $K_{2,3}$



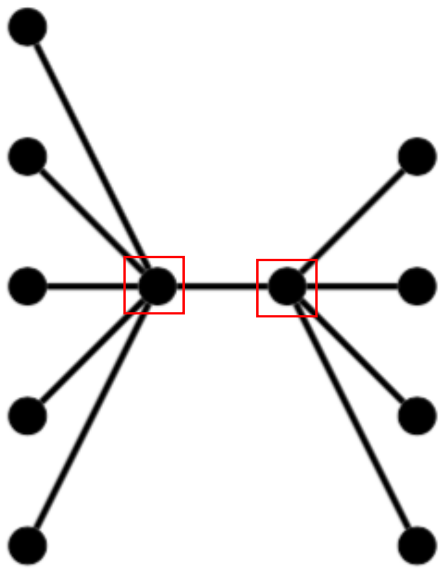
3

$K_{3,3}$

Observation.

A vertex cover VC of a graph G has the following property:

For every vertex v either v or all of its neighbours are contained in VC.



Reductions for VC: Consider an instance *(G=(V,E), k)* of VC

- VC1: If *(G, k)* contains an isolated vertex *v*, delete *v* from *G*; new instance is *(G-v, k)*

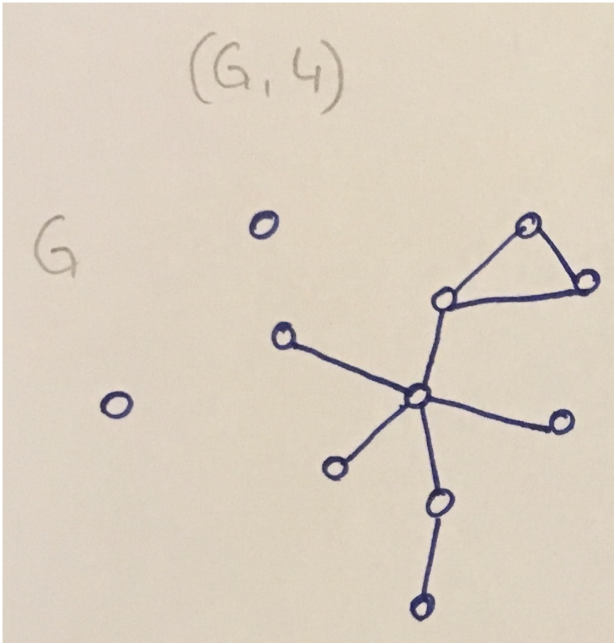
  [safe because *v* does not cover any edge]

- VC2: If *G* has a vertex *v* of degree at least *k+1*, delete *v* from *G* and decrement *k* by one; new instance is *(G-v, k-1)*

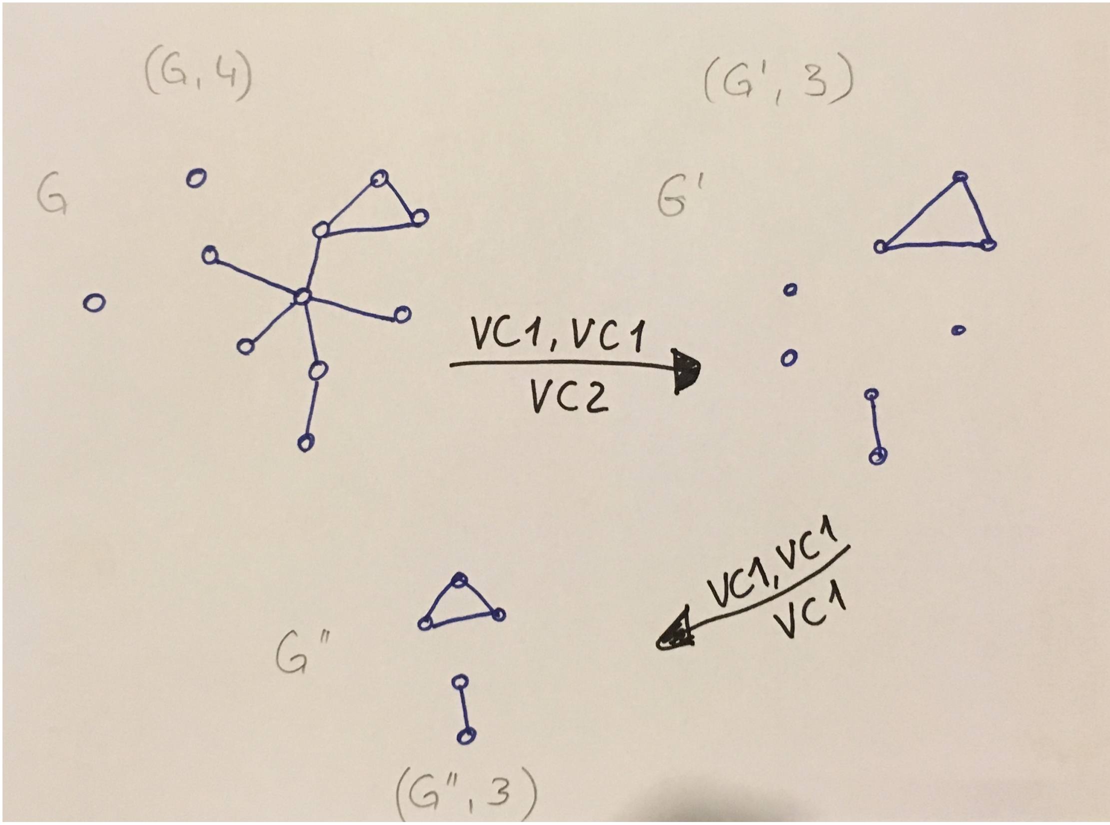
  [safe because *v* must be in VC, otherwise all its at least *k+1* neighbors are in VC; a contradiction (VC cannot have more than *k* vertices)]

- Repeat VC1 and VC2 until no more reduction is possible.

Example 3. Apply VC1 and VC2 to the instance *(G, 4)* until no more reduction is possible.



Example 3. Apply VC1 and VC2 to the instance *(G, 4)* until no more reduction is possible.



Lemma 1. If a graph has maximum degree d, then a set of k vertices can cover at most kd edges.

(Why?) Each vertex covers at most d edges.

Lemma 2. If (G, k) is a yes-instance of VC and neither VC1 nor VC2 can be applied to (G, k), then $|V(G)| \leq k^2+k$ and $|E(G)| \leq k^2$.

Proof.

1. As G is fully reduced under VC2, every vertex of G has at most degree k.
2. As G is a yes-instance, there exists a set of at most k vertices that covers **all** edges of G.
3. Hence, by Lemma 1, we have $|E(G)| \leq k^2$.
4. As G is fully reduced under VC1, there are no isolated vertices in G.
5. As G is a yes-instance, there is a set S of at most k vertices that are (collectively) incident with all edges, i.e. $|E(G)| \leq k^2$. Each edge is incident with a vertex in S and **at most** one vertex not in S.
6. Hence $|V(G)| \leq k+k^2$.

Think about the contrapositive!

Theorem 3. Vertex Cover has a kernel with $O(k^2)$ vertices and $O(k^2)$ edges.

Proof. Follows immediately from Lemma 2.

Input. A graph $G=(V,E)$ and $k \in \mathbb{N}$

Output. Vertex cover C of G of size at most k if it exists.

Set $C=\{\}$.

Set H to be the set of vertices of G whose degree is $> k$.

If $|H|> k$,

then: return "(G,k) is a no-instance"

else: $C := C \cup H$

$j:=k-|H|$

Delete all vertices in H from G and all isolated vertices.

If resulting graph G' has more than j^2 edges or more than j^2+j vertices,

then: return "(G,k) is a no-instance"

else: compute a VC for G' of size at most j (later)

So what?

- Start with a graph G, apply VC1 and VC2 repeatedly until no further reduction is possible. Let G' be the resulting graph. (*How long does this take?*)
- The size of G' only depends on k.
- VC1 and VC2 in combination with a bounded-search tree algorithm (later) gives an algorithm that decides in time $O(n^2+2^k k^2)$ if (G,k) is a yes-instance of VC, where $n=|V(G)|$.
- Hence VC for a graph G is fixed-parameter tractable, when parameterized by the size of a smallest vertex cover for G.
- Running time $O(n^2+2^k k^2)$ can be further improved to $O(1.28^k+nk)$
- Independent of the size of G (e.g. $n=1\,000\,000$), we can now solve VC on a reduced graph with at most k^2 edges.

Kernelization – formal definition

Let L be a parameterized problem consisting of input pairs *(x, k)*. A kernelization algorithm *A* for *L* replaces an instance *(x, k)*, with a reduced instance *(x', k')*, called kernel, such that

- *A* runs in time polynomial in $|x|$
- $k' \leq k$
- $(x,k) \in L$ if and only if $(x',k') \in L$
- $|x'| \leq g(k)$, for some computable function *g* that only depends on *k*

Revisit formal definition: Give VC1 and VC2 a kernelization algorithm?

Let L be a parameterized problem consisting of input pairs *(x, k)*. A kernelization algorithm *A* for *L* replaces an instance *(x, k)*, with a reduced instance *(x', k')*, called kernel, such that

- *A* runs in time polynomial in $|x|$

Reductions VC1 and VC2 can be applied in time $O(n^2)$

- $k' \leq k$

VC1 and VC2 reduce parameter *k* or leave it unchanged

- $(x,k) \in L$ if and only if $(x',k') \in L$

VC1 and VC2 are safe.

- $|x'| \leq g(k)$, for some computable function *g* that only depends on *k*

By Lemma 2., we have $|x'| \leq k^2+k$ vertices.

Theorem. A problem Q is fixed-parameter tractable if and only if it admits a kernelization.

Proof. *[We only prove one implication!]*

Let $I=(x,k)$ be an instance of Q, and let (x',k') be the instance of Q resulting from kernelizing I.

Suppose Q admits a kernelization. Then the reduction can be performed in time $O(|x|^c)$ for some constant c.

Since the size of x' only depends on k and is independent of the input size, we can use brute-force to decide whether or not (x',k') is a yes-instance in time

$$O(|x|^c + f(g(k))).$$

Hence, Q is fixed-parameter tractable.

3-Hitting Set (3-HS)

Instance. A finite set S and a a collection $C=\{C_1,C_2,\ldots,C_m\}$ of subsets of S such that each element in C has size at most three, and a non-negative integer k.

Question. Is there a subset H of S such that $|H|\leq k$ and H contains at least one element from each subset in C?

VC is equivalent to 2-Hitting Set!

3-Hitting Set (3-HS)

Example 4.

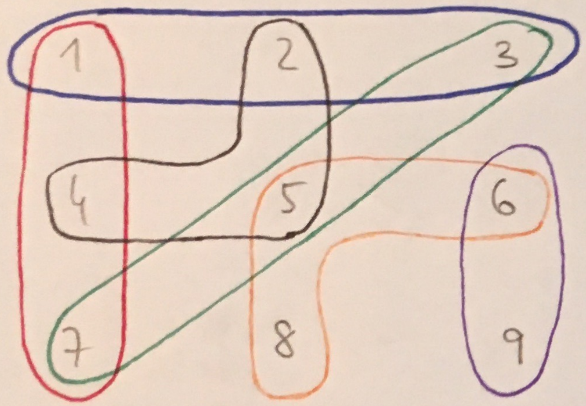
$S=\{s_1,s_2,\ldots,s_9\}$, $k=3$

$C=\{\{s_1,s_2,s_3\}, \{s_1,s_4,s_7\}, \{s_2,s_4,s_5\}, \{s_3,s_5,s_7\}, \{s_5,s_6,s_8\}, \{s_6,s_9\}\}$

Does C have a hitting set H of size at most 3?

$S = \{s_1, s_2, \dots, s_9\}$

$C = \{\{s_1, s_2, s_3\}, \{s_1, s_4, s_7\}, \{s_2, s_4, s_5\}, \{s_3, s_5, s_7\}, \{s_5, s_6, s_8\}, \{s_6, s_9\}\}$



$H_1 = \{s_3, s_4, s_6\}$

$H_2 = \{s_2, s_6, s_7\}$

$H_3 = \{s_1, s_5, s_6\}$

Others ?

Reductions for 3-HS

- HS1. If C contains a singleton $\{s_i\}$, then add s_i to H. Delete $\{s_i\}$ from C. Decrement parameter k by 1.

  s_i must be an element of every hitting set

- HS2. For each pair $C_i, C_j \in C$ with $i \neq j$, if $C_i \subseteq C_j$ delete C_j from C.

  a hitting set that contains an element in C_i also contains an element in C_j

Reductions for 3-HS

- HS3. For each pair $s_i, s_j \in S$ with $i \neq j$, if there are more than k 3-element subsets in C that all contain s_i and s_j, then delete all elements from C that contain s_i and s_j. Add $\{s_i, s_j\}$ to C.

If s_i and s_j not in H, then there are at least $k+1$ other elements in H (*Why?*); hitting set becomes too big.

E.g. $k=3$; if $\{\{1,2,3\},\{1,2,4\},\{1,2,5\},\{1,2,6\}\} \subseteq C$

Once applied exhaustively, there are at most k subsets of the form $\{s_i, s_j, *\}$ in C (otherwise, we can apply HS3 again).

Reductions for 3-HS

- HS4. For each $s_i \in S$, if there are

  more than k 2-element subsets in C or

  more than k^2 3-element subsets in C

  that contain s_i, then add s_i to H. Delete all elements from C that contain s_i. Decrement k by 1.

  If there are more than k subsets of the form $\{s_i, *\}$ or more than k^2 subsets of the form $\{s_i, *, *\}$, add s_i to H; otherwise H becomes too big.

Suppose we are given a yes-instance *(S,C)* of 3-HS and none of the reductions can be applied. What can we say about the size of the instance?

Claim 1. For each pair $s_i, s_j \in S$ with $i \neq j$, there are at most k 3-element subsets in C that have the form $\{s_i, s_j, *\}$.

Proof.

Suppose that there are more than k such sets in C. Let C' be the subset of C that contains all triples of the form $\{s_i, s_j, *\}$. Since C is a set (no element appears multiple times), the third coordinates of all triples in C' are pairwise distinct. Hence, as $|C'|>k$, a hitting set with at most k elements contains s_i or s_j. But then we can replace C' in C with a single element $\{s_i, s_j\}$.

Claim 1 motivates HS3.

Claim 2. For each element $s_i \in S$, there are at most k 2-element subsets in C that have the form $\{s_i, *\}$.

Proof. [Analogous to the proof of Claim 1]

Suppose that there are more than k such sets in C. Let C' be the subset of C that contains all subsets in C of the form $\{s_i, *\}$. Since C is a set (no element appears multiple times), the second coordinates of all elements in C' are pairwise distinct. Hence, as $|C'|>k$, a hitting set with at most k elements contains s_i and we can delete C' from C, decrement k by one and add s_i to H.

Claim 3. For each element $s_i \in S$, there are at most k^2 3-element subsets in C that have the form $\{s_i, *, *\}$.

Proof. Suppose that there are more than k^2 such sets in C. Let C' be the subset of C that contains all triples of the form $\{s_i, *, *\}$. By Claim 1, we may assume that s_i occurs together with s_j in a triple $\{s_i, s_j, *\}$ in C' at most k times.

.

$$\{s_i, s_{j_1}, *\} \leq k \text{ times}$$
$$\{s_i, s_{j_2}, *\} \leq k \text{ times}$$
$$\{s_i, s_{j_3}, *\} \leq k \text{ times}$$
$$\vdots$$
$$\{s_i, s_{j_k}, *\} \leq k \text{ times}$$

Claim 3. For each element $s_i \in S$, there are at most k^2 3-element subsets in C that have the form $\{s_i, *, *\}$.

Proof. Suppose that there are more than k^2 such sets in C. Let C' be the subset of C that contains all triples of the form $\{s_i, *, *\}$. By Claim 1, we may assume that s_i occurs together with s_j in a triple $\{s_i, s_j, *\}$ in C' at most k times. Hence, if $|C'|>k^2$, then the subsets in C' cannot be covered by a hitting set of size at most k that does not contain s_i. Hence s_i is in a hitting set for C and C' can be deleted from C.

Claims 2 and 3 motivate HS4.

Theorem 4. 3-HS has a kernel with $|C| = O(k^3)$ that can be found in polynomial time.

Proof. (based on HS4)

Let $s_i \in S$.

Claim 2 gives us that there are at most k elements in C that have the form $\{s_i,*\}$.

Claim 3 gives us that there are at most k^2 elements in C that have the form $\{s_i,*,*\}$.

Taken together, there are at most $k + k^2$ subsets in C that contain s_i.

Provided C has a hitting set of size at most k, there are at most $k(k + k^2)=k^2+k^3$ elements in C. Hence $|C| = O(k^3)$. (If there are more elements we have a no-instance.)

For a poly-time algorithm, count how many times s_i appears in an element in C.

Theorem 4. 3-HS has a kernel with $|C| = O(k^3)$ that can be found in polynomial time.

What does Theorem 4 say?

If an instance (S,C) of 3-HS has a hitting set of size at most k, then C can be reduced to a set of size $O(k^3)$ by applying the reductions HS1-HS4.

Think about the contrapositive!

If C has more than $O(k^3)$ *elements after applying the reductions* HS1-HS4, then (S,C) is a no-instance. The algorithm can stop and return `no' without performing any exhaustive search.

Generalization to *d*-Hitting Set

Instance. A finite set S and a collection $C=\{C_1,C_2,\ldots,C_m\}$ of subsets of S such that each element in C has size at most d, and a non-negative integer k.

Question. Is there a subset H of S such that $|H|\leq k$ and H contains at least one element from each subset in C?

Theorem 4. For a **fixed** $d>3$, d-HS has a kernel with $|C|= O(k^d)$.