Constructive Proof of the Lovasz Local Lemma

Katharina Angermeier

November 3, 2014

Contents

1 Introduction

In the introduction some notations and definitions we will require for the presentation will be explained. The *conjunctive normal form* (CNF) is a special notation form for boolean formulas. An example would be the following 3-CNF formula with 4 clauses over the set of variables $\{x_1, x_2, x_3, x_4\}$:

$$(x_1 \vee x_2 \vee x_3) \wedge (x_1 \vee x_3 \vee \overline{x_4}) \wedge (\overline{x_1} \vee \overline{x_2} \vee x_4) \wedge (x_2 \vee \overline{x_3} \vee x_4)$$

Variables in a clause do not repeat. In general we define a k-CNF formula ($k \in \mathbb{N}$) as a CNF formula where every clause contains exactly k literals. An *assignment* α over a variable set V is a mapping $\alpha : V \to \{0, 1\}$, that extends to $\overline{V}$ via $\alpha(\overline{x}) := 1 - \alpha(x)$ for $x \in V$.

A formula is called *satisfiable* if there is a true-false assignment to the variables so that every clause has at least one literal that evaluates to true. In our example, a satisfying assignment could be $(x_1, x_2, x_3, x_4) \mapsto (true, true, false, true)$. We define $\mathrm{vbl}(C)$ as the set of variables that occur in a clause C. And equally $\mathrm{vbl}(F) := \bigcup_{C \in F} \mathrm{vbl}(C)$ for F a CNF formula.
We can already make some statements to satisfiability.

Claim: *It takes at least 2^k clauses to construct an unsatisfiable k-CNF formula.*

Justification: Suppose some k-CNF formula with fewer than 2^k clauses. An assignment sampled uniformly at random violates each clause with probability 2^{-k}. So by linearity of expectation we can say that the expected total number of violated clauses is smaller than 1. That means that there needs to be at least one assignment that satisfies the whole formula.

Now it is easy to see that if some of the clauses are independent from each other we need more than 2^k clauses to obtain an unsatisfiable k-CNF formula. So the constraint on the formula size needs not only to be satisfied globally but even locally. To observe that we introduce the *neighbourhood* $\Gamma(C) = \Gamma_F(C) := \{D \in F \mid vbl(D) \cap vbl(C) \neq \emptyset\}$ of a clause C, which is the set of clauses that share variables with C. By intuition we can see that if we can change values in a clause C without causing too much damage in its neighbourhood, and if this property holds everywhere, then maybe we can find a globally satisfying assignment by just moving around violation issues.

> *If every clause in a k-CNF formula, $k \geq 1$, has a neighbourhood of size at most $2^k/e - 1$, then the whole formula admits a satisfying assignment.*
> *Lovász Local Lemma*, 1975

> Other variant: *"In an unsatisfiable CNF formula clauses have to interleave - the larger the clauses, the more interleaving is required."*

So this is the main theorem the presentation is about. The big importance of that lemma is not really easy to see in the formulation above, but if we observe the clauses as any collection of events in a probability space, the lemma becomes very general. It is the so called symmetric form of the lemma.
We define the *conflict-neighbourhood* $\Gamma'(C) = \Gamma'_F(C) := \{D \in F \mid C \cap \overline{D} \neq \emptyset\}$ of a clause C as the set of clauses which share variables with C, at least one with opposite sign. With that definition we can "measure" the quality of interleaving. The so called *lopsided Local Lemma* shows the condition for neighbourhoods holds actually for conflict-neighbourhoods.
The *degree of* x is the number of occurrences of a variable x (with either sign) in a CNF formula, $deg(x) = deg_F(x) := |\{C \in F \mid x \in vbl(C)\}|$.

Claim: *If every variable in a k-CNF formula, $k \geq 1$, has degree at most $2^k/(ek)$, then the formula is satisfiable.*

For formulas with little interleaving we define a *linear CNF formula* as a CNF formula where any two clauses share at most one variable.

Example: $(\overline{y_1} \vee \overline{y_2}) \wedge (y_1 \vee x) \wedge (y_2 \vee x) \wedge (z_1 \vee \overline{x}) \wedge (z_2 \vee \overline{x}) \wedge (\overline{z_1} \vee \overline{z_2})$
This is a smallest unsatisfiable linear 2-CNF formula.

Claim: *Any linear k-CNF formula with at most $4^k/(4e^2k^3)$ clauses is satisfiable.*

Whenever the easily checkable conditions formulated above are satisfied, then the algorithmic problem of deciding satisfiability becomes trivial. However, the actual construction of a satisfying assignment is by no means obvious.

We define $f(k)$, $k \in \mathbb{N}$, as the largest integer so that every k-CNF formula with no variable of degree exceeding $f(k)$ is satisfiable. We know that $f(k) = \Theta(2^k/k)$. Although $f(k)$ is not known for k exceeding 4 one can show a sudden jump behaviour in complexity. For k-CNF formulas ($k \geq 3$) with max-degree at most $f(k)+1$ the satisfiability problem becomes NP-complete. A similar immediate transition can be observed for the related problem for the conflict-neighbourhood size. $l(k)$ is defined as the largest integer d such that every k-CNF formula F for which $|\Gamma_F(C)| \leq d$, for all $C \in F$, is satisfiable. $lc(k)$ is defined analogously, but with $|\Gamma'_F(C)| \leq d$.

In some of the proofs we use the relation of CNF formulas to hypergraphs. A *hypergraph* H is a pair (V, E) with V a finite set and $E \subseteq 2^V$. It is *k-uniform* if $|e| = k$ for all $e \in E$. H is called *2-colourable* if there is a colouring of the vertices in V by two colors red and green so that no hyperedge in E is monochromatic. The relation to satisfiablility of CNF formulas is that $H = (V, E)$ is 2-colourable iff the CNF formula $E \cup \{\overline{e} \mid e \in E\}$, with V now considered as set of boolean variables, is satisfiable.

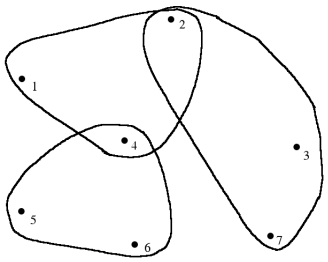

Figure 1: 3-uniform hypergraph

The corresponding formula to Figure 1 is:
$(x_1 \vee x_2 \vee x_4) \wedge (\overline{x_1} \vee \overline{x_2} \vee \overline{x_4}) \wedge (x_2 \vee x_3 \vee x_7) \wedge (\overline{x_2} \vee \overline{x_3} \vee \overline{x_7}) \wedge$
$(x_4 \vee x_5 \vee x_6) \wedge (\overline{x_4} \vee \overline{x_5} \vee \overline{x_6})$

Figure 2 would be an assignment such that the formula evaluates to true, no matter if red is considered as true and green as false or the other way round.

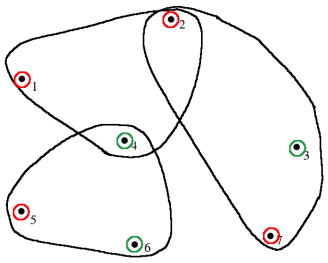

Figure 2: 2-coloured 3-uniform hypergraph

2 Local Lemma in Terms of SAT - Proof and Algorithm

Theorem 1 *Let $k \in \mathbb{N}$ and let F be a k-CNF formula. If $|\Gamma(C)| \leq 2^k/e - 1$ for all $C \in F$, then F is satisfiable.* P. Erdős, L. Lovász: Problems and results on 3-chromatic hypergraphs and some related questions.

Theorem 2 (Lovász Local Lemma, symmetric form) *Let $A = A_1, A_2, \ldots, A_m$ be any collection of events in a probability space, each one having probability at most p and such that each event is mutually independent of all but at most d of the other events. If $ep(d+1) \leq 1$, then with positive probability, none of the events in A occur.*

The SAT formulation, Theorem 1, follows as an immediate corollary. Considering the random experiment of sampling truth assignments to the CNF formula F at random and defining A_i to be the event that clause number i becomes violated, each event has probability 2^{-k} and the desired bound follows. This way, it is a natural extension of the simple pobabilistic argument bounding from below the total number of clauses in an unsatisfiable formula.

The first "existential" proof, which was given in 1975, was short but non-constructive. Then, for a long time nothing happend in that area until in 1991 Beck proved the existence of a polynomial-time algorithm to find a satisfying assignment for all $C \in F$, F a k-CNF formula $\Gamma(C) \leq 2^{k/48}$. In the same year Alan simplified Beck's algorithm by randomness, and presented an algorithm that works for neighbourhoods of size up to $2^{k/8}$. About ten years later Czumaj and Scheideler demonstrated that a variant of the method can be made to work for the case where clauses sizes vary. The actual breakthrough came 2008 when Moser published an polynomial-time algorithm for neighbourhood sizes up to $O(2^{k/2})$, later for 2^{k-5} neighbours. In 2009 Moser and Tardos published a fully constructive proof.

2.1 First Proof of Local Lemma - Existence

Since the existencial proof is based on a probabilistic argument, for its comprehension it is helpful to observe some simple examples of the probability that a random assignment satisfies a CNF-formula. There are some minimal examples in Figure 3.

Let F be a k-CNF formula with neighbourhood size at most $d := \frac{2^k}{e} - 1$ other clauses. The main idea of the proof is that if the probability of a random assignment α to satisfy F is positive, F is satisfiable. Let $F' \subset F$ be any subformula of F with one fewer clause. Let $C \in F \backslash F'$ be one of the clauses removed. The assignment α has a certain probability $Pr(F')$ of satisfying F'. We now want to compute the drop in probability when adding back C as an additional constraint.
We claim that the drop is bounded by a factor of $(1 - e2^{-k})$, which means $Pr(F' \wedge C) \geq (1 - e2^{-k})Pr(F')(\Leftrightarrow Pr(F' \wedge \neg C) \leq e2^{-k}Pr(F'))$. If the factor is positive, the claim is proved, since the

$k=1$	p
$\emptyset$	1
x_1	$\frac{1}{2}$
$x_1 \wedge x_2$	$\frac{1}{4}$
$x_1 \wedge x_2 \wedge x_3$	$\frac{1}{8}$
$x_1 \wedge x_2 \wedge x_3 \wedge x_4$	$\frac{1}{16}$

$k=2$	p
$\emptyset$	1
$(x_1 \vee x_2)$	$\frac{3}{4}$
$(x_1 \vee x_2) \wedge (x_3 \vee x_4)$	$\frac{9}{16}$
$(x_1 \vee x_2) \wedge (x_3 \vee x_4) \wedge (x_5 \vee x_6)$	$\frac{27}{64}$

Figure 3: Probability p of random assignments to satisfy a k-CNF formula

empty formula is satisfied with probabilitiy 1 and then successively adding back all of F's clauses diminishes that probability by a positive factor each step, leaving a positive probabiliy in the very end.

We proceed inductively. Suppose the latter claim has been proved for all subformulas F' up to a given size and now we would like to establish it for larger subformulas. There is a trivial special case. If the constraint C that we join back to F' is independent, so has no variables in common with F', the probability decreases by a factor of exactly $(1-2^{-k})$. We can see that factor in the tabulars in Figure 3. Now we have to show why lowering that factor to $(1-e2^{-k})$ is sufficient to account for the amount of possible dependencies that we might encounter. So if C shares some variables with F', we remove all clauses of F' neighbouring C and get $F'' := F' \backslash \Gamma(C)$ to get rid of those dependencies. Now F'' and C are independent.

$$\Rightarrow Pr(F'' \wedge \neg C) = 2^{-k} Pr(F'')$$

By adding back all clauses one by one to F'' to get F' we obtain

$$Pr(F') \geq (1-e2^{-k})^d Pr(F'') \geq e^{-1} Pr(F'')$$

Now since every assignment satisfying F' satisfies F'', we have

$$Pr(F' \wedge \neg C) \leq Pr(F'' \wedge \neg C) = 2^{-k} Pr(F'')$$

$$\Rightarrow \frac{Pr(F' \wedge \neg C)}{Pr(F')} \leq \frac{2^{-k}}{e^{-1}}$$

2.2 Second Proof of Local Lemma - Algorithm

The main idea of the Algorithm is: We repeatedly select any of the violated clauses and just select new uniformly random variables occurring in that clause until a satisfying assignment is obtained.

Now we want to show that under the hypothesis of the Local Lemma it converges to a satisfying assignment in an expected polynomial number of steps. For the analysis of the algorithm we record a log of corrections with the mapping $L : \mathbb{N}_0 \to F$. In step t, the algorithm selects clause $L(t)$ for correction. We hope for the algorithm to terminate after a a finite number of steps, but for the moment we have to allow for an infinite log and then prove that we will never encounter one. Let $N : F \to \mathbb{N}_0 \cup \{\infty\}$ be random variables that count the number of times a given clause occurs in the log. So $N(C) := |\{t \in \mathbb{N}_0 \mid L(t) = C\}|$. Again we have to allow for such a counter to take infinity as a value. We prove now that for each clause $C \in F$ the expected value $E[N(C)]$ is upper bounded by a constant, so every clause is corrected at most a constant number of times. That means the total number of clauses corrected is bounded by $O(|F|)$.

To continue we introduce witness trees. A witness tree is an unordered, rooted tree T along with a labelling $\sigma : V(T) \to F$ of its vertices $V(T)$ by clauses from F. For every time index t such that $L(t)$ is defined we build a witness tree in the following sense. We label the root vertex r $\sigma(r) := L(t)$. Now we traverse the log backwards and for each time step $s = t-1, t-2, \ldots, 0$, check if the clause $L(s)$ has any variables that it shares with any of the labels in the tree built so far. If $L(s)$ is independent from all clauses currently serving as labels, we discard it. Otherwise we select any deepest of the nodes the tree has wich have variables in common with $L(s)$ and create a new child node of it, labelling that new child $L(s)$. When arriving at $s = 0$ we have built a witness tree $T(t)$ that justifies correction step t.
We can reconstruct a significant portion of the execution history only by looking at the witness tree $T(t)$. By traversing $T(t)$ in a breadth-first-search that starts at the root we obtain a sequence of clauses that is a subsegment of the execution log. Each node we encounter represents some correction step in L with the label of the node being the clause corrected in that step.
The way we defined $T(t)$ assures two things:
(a) The ordering in which the corrections have taken place is similar to the ordering in which we traverse the nodes, in the following sense: Whenever two nodes v_1 and v_2 are labelled with clauses that depend on each other, then v_1 occurs before v_2 in the traversal if and only if v_1 represents a correction step occurring before v_2.
(b) When we traverse some node v representing correction step t, then all correction steps $t' < t$ that relate to step t, in the sense that $L(t)$ and $L(t')$ share common variables, do occur in the tree and have therefore been traversed before.
These two properties imply that the number of times some variable x has occurred so far in labelling clauses corresponds to the number of times x has been reassigned new values before the corresponding correction step.

So if we have seen variable x already 10 times before we traverse a node v labelled $\sigma(v) = C$, then this means that at the time the correction v represents took place, x had its 10th new random value and was then assigned its 11th one. This in turn means that we can reconstruct, by just looking at the tree, all the 10 values x had been assigned before. This is because node v represents a time step where clause C was selected for correction, that is a time step when C was violated and thus the 10th value of x has to have been the one hat dissatisfies the corresponding literal we find in C. The same holds for all other variables in the clause and for all other nodes we traverse.

Now what about when you have given a fixed witness tree T, what is the probability that exactly this tree can occur as witness for some correction step? We can reconstruct for each node the values the k variables in the corresponding clause were assigned before the correction step represented, that is we can reconstruct k of the random bits the algorithm has used. If the tree has n vertices, we can reconstruct nk bits in total, just by looking at the tree. The probability that all of them sample such that T can be constructed is exactly 2^{-nk}. For a fixed clause $C \in F$, number n, we want the number of witness trees of order n which have C as the label of their root vertex. That number is restricted by the definition of witness trees, which requires that if u is a child node of v, then the label $\sigma(u)$ must be a neighbour of the clause $\sigma(v)$. This allows us to embed each witness tree rooted at label C into an infinite tree that just enumerates neighbouring nodes. Consider an infinite tree with its root labelled C and such that each node v labelled $\sigma(v)$ has $|\Gamma(\sigma(v))|$ children labelled $\Gamma(\sigma(v))$. Such a tree is at most $(\leq d)$-ary and each witness tree is clearly a subtree of it. An infinite rooted $(\leq d)$-ary tree has at most $(ed)^n$ subtrees of size n. That implies that there are at most $(ed)^n$ witness trees of order n that have C as their root label. The expected number of witness trees of size n that can occur is bounded by $(ed2^{-k})^n$, since each of them may occur with a probability of at most 2^{-nk}. Summing over all possible sizes $n \geq 1$ this becomes a geometric series that converges to a constant, so there is at most a constant expected number of valid witness trees

rooted at C.

For each of the $N(C)$ $(t_1, t_2, \ldots, t_{N(C)})$ times a clause C occurs in the execution log we can ask for a corresponding witness tree $T(t_1), T(t_2), \ldots T(t_{N(C)})$ to justify that correction step. The trees are distinct since $T(t_{i+1})$ needs to have basically the same vertices as $T(t_i)$ and at least one more (for step t_{i+1}). $N(C)$ is at most as large as the number of valid witness trees rooted at C, which is bounded by a constant in expectation. □

2.3 A Stronger Variant - Conflicts

The so called lopsided Local Lemma, which does not only distinguish between dependent and independent events but also discriminates between positive and negative correlations, is a slightly stronger version of the Lovász Local Lemma. So the bound on the maximum neighbourhood size is replaced by a bound on conflict neighbourhoods.

Theorem 3 *Let $k \in \mathbb{N}$ and let F be a k-CNF formula. If $|\Gamma'(C)| \leq 2^k/e - 1$ for all $C \in F$, then F is satisfiable.*

Both proofs for the Lovász Local Lemma can be adapted to demonstrate this statement. For the constructive proof the same algorithm will work and for the analysis it suffices to observe that witness trees built by attaching only lopsided neighbours during backward traversal of the log equally allow to reconstruct k bits of the randomness used per vertex, irrespective of the fact that a smaller amount of information might be encoded by the tree. Berman, Karpinski and Scott have demonstrated using the lopsided Local Lemma, that every 6-, 7-, 8- or 9-CNF formula in which every variable occurs at most 7, 13, 23 or 41 times, respectively, is satisfiable.

3 Bounded Variable Degree

A k-CNF formula in which no variable occurs in more than d clauses is called a (k, d)-CNF formula. So $f(k)$ is now defined as the unique integer so that all $(k, f(k))$-CNF formulas are satisfiable and an unsatisfiable $(k, f(k)+1)$-CNF formula exists. We know that $f(k)$ exists with $0 \leq f(k) \leq 2^k$. Tovey was the first to consider $f(k)$ in 1984. He showed $f(k) \geq k$ and conjectured that all $(k, 2^{k-1} - 1)$-CNF formulas are satisfiable. A clearer picture of $f(k)$ has evolved since then. For, if every variable occurs at most d times in a k-CNF formula, no clause can have more than $k(d-1)$ neighbours. Thus, $k(d-1) \leq 2^k/e - 1$ implies that every (k, d)-CNF formula is satisfiable. This connection was made by Kratochvíl, Savický and Tuza, who also established 1993 the bounds of $f(k) \geq \lfloor 2^k/(ek) \rfloor$ and $f(k) \leq 2^{k-1} - 2^{k-4} - 1$. This is still the best lower bound known for k large. Another significant progress on the upper end was made by Savický and Sgall, when they showed $f(k) = O(k^{-0.26}2^k)$ (2000). Hoory and Szeider improved it to $f(k) = O((2^k log k)/k)$ (2006). Recently Gebauer settled $f(k) = \Theta(2^k/k)$.

Theorem 4 *For k a large enough integer,*

$$\lfloor 2^k/ek \rfloor \leq f(k) < 2^{k+1}/k.$$

If k is a sufficiently large power of 2 we have $f(k) < 2^k/k$.

To demonstrate that SAT connects to many (sometimes seemingly unrelated) problems, I explain you the main idea of the proof of the upper bound in this Theorem. The actual construction was originally developed for refuting a conjecture of Beck on Combinatorial games. In such a game

we have a Maker and Breaker who take turns in choosing vertices from a given hypergraph. Maker wants to completely occupy a hyperedge and Breaker tries to prevent this. The problem is to find the minimum $d = d(k)$ such that there is a k-uniform hypergraph of maximum vertex degree d where Maker has a winning strategy. If Maker uses a pairing strategy, which means he partitions all the vertices and if Breaker claims one vertex of a pair, Maker takes the other one, this game is equivalent to unsatisfiability. A hypergraph H, pairing P can be interpreted as a CNF formula F where the hyperedges of H are clauses and two vertices of a pair of P are complementary literals. Maker wins the game on H using the pairing strategy according to P if and only if F is unsatisfiable.
If there is a k-uniform hypergraph of maximum vertex degree d with a winning pairing strategy for Maker, then there is an unsatisfiable $(k, 2d) - CNF$ formula.

3.1 Small Values

Although the lower bounds on $f(k)$, $l(k)$ and $lc(k)$ we can derive via the Local Lemma grow exponentially, they are weak for small values of k.

Lemma 3 *(1) $f(k) \geq k$ for $k \geq 1$ and (2) $l(k) \geq lc(k) \geq k$ for $k \geq 2$*

Sketch of proof of (1): For $k \geq 1$, let F be a k-CNF formula over a variable set V, no variable occurring in more than k clauses. Consider the incidence graph between clauses and variables, which is a bipartite graph with vertex set $F \cup V$, where $\{C, x\}$ is an edge iff $x \in vbl(C)$. In this graph, clause-vertices have degree exactly k and by assumption variable-vertices have degree at most k.
So Hall's condition for a matching covering all clause-vertices holds. An assignment is now defined by letting every variable x that is matched to a clause C map to the value so that it satisfies C. The matching property prevents conflicts and no matter how we complete the assignment for unmatched variables it will satisfy all clauses. $\Rightarrow$ (1).

$f(k) = k$ is known for $k \leq 4$, the best known bounds for $k = 5$ are $5 \leq f(5) \leq 7$. $k = 6$ is the first value for which the bound in Lemma 3(1) is known not to be tight: $7 \leq f(6) \leq 11$.

4 Linear Formulas

A CNF formula F is called linear if $|vbl(C) \cap vbl(D)| \leq 1, \quad C, D \in F, C \neq D$. For example, the formula $\{\{x, y\}, \{\overline{y}, z\}, \{\overline{x}, \overline{z}\}\}$ is linear. This class of formulas is a natural analogue of the notion of *linear hypergraphs*: A hypergraph $H = (V, E)$ is linear if $|e \cap f| \leq 1$ for any two distinct edges $e, f \in E$.

Given a k-uniform non-2-colorable hypergraph H with m hyperedges, we immediately obtain an unsatisfiable k-CNF formula $F(H)$ with $2m$ clauses. For $k \geq 2$, even if H is linear, $F(H)$ is certainly not. So it is not clear that bounds on the size of unsatisfiable linear k-CNF formulas are similar to those of non-2-colourable linear k-uniform hypergraphs.

Let $f_{lin}(k)$ be the largest integer so that every linear $(k, f_{lin}(k))$-CNF formula is satisfiable. Note that $f_{lin} \geq f(k) \geq \lfloor 2^k/(ek) \rfloor$.

Theorem 6 *Any unsatisfiable linear k-CNF formula has at least*

$$\frac{1}{k}(1 + f_{lin}(k-1))^2 > \frac{4^k}{4e^2k^3}$$

clauses. There exists an unsatisfiable linear k-CNF formula with at most $8k^3 4^k$ clauses.
Remark. $\frac{1}{k}(1 + f_{lin}(k-1)^2) \leq 8k^3 4^k$ follows thus $f_{lin}(k-1) \in O(k^2 2^k)$.

Idea of the proof. The proof is similar to the proof for the size of non-2-colourable linear k-uniform hypergraphs in "Problems and results on 3-chromatic hypergraphs and some related questions" (Erdős, Lovász).

Lemma 5 *Let F be a linear k-CNF formula. If there are at most $f_{lin}(k-1)$ variables of degree exceeding $f_{lin}(k-1)$, then F is satisfiable.*

Let X be the set of variables x with $deg_F(x) > f_{lin}(k-1)$. If F is unsatisfiable $|X| > f_{lin}(k-1)$. Therefore the lower bound follows from

$$|F| = \sum_{x \in vbl(F)} deg_F(x) \geq \frac{1}{k}(1 + f_{lin}(k-1))|X| \geq \frac{1}{k}(1 + f_{lin}(k-1))^2$$

The whole proof will not be done here since it would be very long and it is not really important.

5 A Sudden Jump in Complexity

Although satisfiability of $(k, f(k))$-CNF formulas is trivially decidable in polynomial time, if we relax the bound of the degree it becomes NP-complete. Tovey proved in 1984 that for 3-CNF formulas with maximum variable degree $f(3) + 1 = 4$ satisfiability is NP-complete. Later (1993) Kratochvíl, Savický and Tuza generalised this sudden jump: For every fixed $k \geq 3$, satisfiability of $(k, f(k) + 1)$-CNF formulas is NP-complete. It may be somewhat intriguing that one can prove such a result, given that we do not even know the values of $f(k)$ for $k \geq 5$; but we will see. Berman, Karpinski and Scott (2003) showed that for $(k, f(k) + 1)$-CNF formulas it is even hard to approximate the maximum number of clauses that can be simultaneously satisfied. We will approach the related problems for the size of neighbouhoods and conflict-neighbourhoods. While we can show that the latter performs a similar sudden jump, we have to leave a slack for the neighbourhood bound.

Theorem 9 *Let $k \geq 3$. Then,*
(1) deciding satisfiability of k-CNF formulas with variable degrees at most $f(k)+1$ is NP-complete
(2) deciding satisfiability of k-CNF formulas with clause neighbourhoods of size at most $max\{k + 3, l(k) + 2\}$ is NP-complete
(3) deciding satisfiability of k-CNF formulas with clause conflict-neighbourhoods of size at most $lc(k) + 1$ is NP-complete

For the proof we describe a general construction that takes a k-CNF formula F and produces a CNF formula $\hat{\mathrm{F}}$ which is satisfiable iff F is satisfiable, so that $\hat{\mathrm{F}}$ is very sparsely interleaved, at the expense of the appearance of 2-clauses. We will later expand these 2-clauses to k-clauses in a fashion tailored to which of the three claims we want to prove. Given a set of $j \geq 2$ variables, $U = \{x_0, x_1, \ldots, x_{j-1}\}$, the 2-CNF formula

$$\{\{x_0, \overline{x_1}\}, \{x_1, \overline{x_2}\}, \ldots, \{x_{j-2}, \overline{x_{j-1}}\}, \{x_{j-1}, \overline{x_0}\}\}$$

is called an equaliser of U. The equaliser of a singleton set U is the empty formula. It is easy to see that such an equaliser is satisfied by an assignment to U iff all variables in U are mapped to the same value. Let F be a k-CNF formula, $k \geq 3$. For each variable $x \in$vbl(F), we replace every occurrence by a new variable inheriting the sign of x in this occurrence. This yields a k-CNF formula F' with $|F|$ clauses over a set of $k|F|$ variables. For each $x \in vbl(F)$ we add an equaliser for the set of variables that have replaced occurrences of x. This gives a set F'' of at most $k|F|$ 2-clauses. By the property of equalisers, $\hat{\mathrm{F}} := F' \cup F''$ is satisfiable iff F is satisfiable. $\hat{\mathrm{F}}$ can be obtained from F in polynomial time. We know that every variable of vbl$(\hat{\mathrm{F}})$ occurs at most 3 times in $\hat{\mathrm{F}}$. It is also given that each k-clause in F' does not share variables with any other clause in F'

and the number of its neighbouring 2-clauses in F'' is at most $2k$ and at most k of the 2-clauses are in the conflict-neighbourhood. Another thing we know is that each 2-clause in F'' neighbours two k-clauses in F' and at most two 2-clauses in F''.
With that construction we can prove the theorem.

Proof of (1) (variable degrees) Let $k \geq 3$ and fix some minimal unsatisfiable $(k, f(k)+1)$-CNF formula G. We choose some clause C in G and replace one of its literals by $\overline{x}$ for a new variable x to get $G(x)$. This new formula $G(x)$ is satisfiable (otherwise G would not be minimal), every satisfying assignment has to set x to 0 (since otherwise G would be satisfiable), all variables have degree at most $f(k)+1$ and $deg_{G(x)}(x) = 1$.
Given a k-CNF formula F we first generate $\hat{\mathrm{F}}$, as described before the proof. Then we augment each 2-clause in $\hat{\mathrm{F}}$ by $(k-2)$ positive literals of new variables so that it becomes a k-clause. For each new variable x we add a copy of $G(x)$ to our formula. By renaming variables in G these copies are chosen so that their variable sets are pairwise disjoint. By construction, the new formula is satisfiable iff $\hat{\mathrm{F}}$ is satisfiable. The maximum variable degree is $max\{3, f(k)+1\}$, which is $f(k)+1$, since $k \geq 3$.
This constitutes a polynomial reduction of satisfiability of general k-CNF formulas to satisfiability of k-CNF formulas with maximum variable degree $f(k)+1$. □

The proof of (2) and (3) is very similar to the proof of (1), so it will not be done here.

6 Open Problems

Now there will be mentioned some open problems in that area. We know $f(k)$, $l(k)$ and $lc(k)$ up to a constant, so one might hope to eventually determine them exactly. Progress on the lower bounds would also be very interesting.
Open Problem 1. *Is it possible to improve any of the known lower bounds on $f(k)$, $l(k)$, and $lc(k)$ by a constant factor?*
For that one possible approach would be to better understand how these functions depend on each other. For example, the current lower bound on $f(k)$ follows by a very simple argument from a lower bound on $l(c)$.
Open Problem 2. *Is there a constant $c_0 > 1$ with $f(k) \geq c_0 l(k)/k$ for k large enough?*
Open Problem 3. *Is there a constant $c_1 > 1$ such that $l(k) \geq c_1 lc(k)$ for k large enough?*
Open Problem 4. *Are the functions $f(k)$, $l(k)$ and $lc(k)$ computable?*