and $\pi(2) = j$. We claim that $|S_{12}| \leq |S_{ij}|$ for all $i \neq j$. Indeed, suppose first that $i, j > 2$. For each good $\pi \in S_{12}$ define a permutation π^* as follows. Suppose $\pi(x) = i$, $\pi(y) = j$. Then define $\pi^*(1) = i, \pi^*(2) = j, \pi^*(x) = 1, \pi^*(y) = 2$ and $\pi^*(t) = \pi(t)$ for all $t \neq 1, 2, x, y$. One can easily check that π^* is good, since the cells $(1, i)$, $(2, j)$, $(x, 1)$, $(y, 2)$ are not part of any $(p, q, p', q') \in S$. Thus $\pi^* \in S_{ij}$, and since the mapping $\pi \to \pi^*$ is injective $|S_{12}| \leq |S_{ij}|$, as claimed. Similarly one can define injective mappings showing that $|S_{12}| \leq |S_{ij}|$ even when $\{i, j\} \cap \{1, 2\} \neq \emptyset$. It follows that

$$\Pr\left[A_{1122} \wedge \bigwedge_S \overline{A_{pqp'q'}}\right] \leq \Pr\left[A_{1i2j} \wedge \bigwedge_S \overline{A_{pqp'q'}}\right]$$

for all $i \neq j$ and hence that

$$\Pr\left[A_{1122} \mid \bigwedge_S \overline{A_{pqp'q'}}\right] \leq \frac{1}{n(n-1)}\,.$$

By symmetry, this implies (5.8) and completes the proof. ■

5.7 MOSER'S FIX-IT ALGORITHM

When the probabilistic method is applied to prove that a certain event holds with high probability, it often supplies an efficient deterministic, or at least randomized, algorithm for the corresponding problem.

By applying the Local Lemma we often manage to prove that a given event holds with positive probability, although this probability may be exponentially small in the dimensions of the problem. Consequently, it is not clear if any of these proofs can provide polynomial algorithms for the corresponding algorithmic problems. For many years there was no known method of converting the proofs of any of the examples discussed in this chapter into an efficient algorithm. In 1991 József Beck found such a method that works for some of these examples, with a little loss in the constants. This has been extended and modified by several researchers. In 2009 Robin Moser found a remarkably simple algorithm, combined with a subtle analysis. This approach has been extended in his joint work with Tardos which provides an efficient algorithm for essentially all known applications of the Local Lemma, with no loss in the constants. In fact, in some examples the constants obtained are even better than those that follow from the earlier existence proofs. We proceed with the details, following Moser and Tardos (2010).

We first give the context. Let Ω be a finite set. For each $v \in \Omega$ let $C[v]$ denote a random variable. The variables $C[v]$ may have different distributions but, critically, they are mutually independent. Ω and the $C[v]$ then define a probability space. Let I denote an index set. For each $\alpha \in I$ there is an associated set $A[\alpha] \subset \Omega$ and an event $BAD[\alpha]$. Set $p[\alpha] = \Pr[BAD[\alpha]]$ Critically, the event $BAD[\alpha]$ depends

only on the values $C[v]$ for $v \in A[\alpha]$. Our goal shall be to find (under suitable side conditions) specific values for $C[v]$ such that none of the $BAD[\alpha]$, $\alpha \in I$, hold. We define $\sim$ on I by setting $\alpha \sim \beta$ if $A[\alpha] \cap A[\beta] \neq \emptyset$. (Note that $\alpha \sim \alpha$.) The relation $\sim$ does yield a dependency graph D on the events $BAD[\alpha]$ with (α, β) an edge if $\alpha \sim \beta$. The event $BAD[\alpha]$ is mutually independent of all $BAD[\beta]$ with (α, β) not an edge since those events are determined by $C[v]$ for $v \notin A[\alpha]$.

As an instructive example, let $A[\alpha] \subset \Omega$ be a family of k element sets. For $v \in \Omega$ let $\Pr[C[v] = Red] = \frac{1}{2}$ and $\Pr[C[v] = Blue] = \frac{1}{2}$. That is, C is a random 2-coloring of Ω. For each α let $BAD[\alpha]$ denote the event that $A[\alpha]$ is monochromatic. Thus all $p[\alpha] = 2^{1-k}$. Our goal is then to find (under suitable side conditions) a coloring of Ω for which no $A[\alpha]$ is monochromatic. In other cases Ω consists of Boolean variables $x_1, \ldots, x_n$ and $C[x_i]$ is true or false with independent probability one half. Another type: $S \subset \Omega$ is a random set, $C[v]$ is that $v \in S$, and the $C[v]$ are determined by independent, though not necessarily identical, coin flips. When Ω is the set of pairs $\{i, j\} \subset \{1, \ldots, n\}$ and $\Pr[C(\{i, j\}) = 1] = p_{i,j}$, $\Pr[C(\{i, j\}) = 0] = 1 - p_{i,j}$, one gets a broad generalization of $G(n, p)$. It is even possible for $C[v]$ to be multi-valued. Indeed, essentially all applications of the Local Lemma can be placed in this context.

MOSER's FIX-IT ALGORITHM

FIX-IT I: For each $v \in \Omega$ choose $C[v]$ according to its distribution.

FIX-IT II: WHILE at least one $BAD[\alpha]$ holds

FIX-IT III: Select one $\alpha \in I$ for which $BAD[\alpha]$.

FIX-IT IV: Reset $C[v]$ for each $v \in A[\alpha]$.

END WHILE

Tautologically, when and if the FIX-IT Algorithm terminates the desired values of $C[v]$ have been found. FIX-IT III allows an arbitrary deterministic selection process, for definiteness we may imagine I to be linearly ordered and select the first $\alpha \in I$ for which $BAD[\alpha]$. We define the LOG to be the sequence $\alpha_1 \alpha_2 \cdots \alpha_u$ where α_t is the α selected (*not* all the α with $BAD[\alpha]$!) the t-th time FIX-IT III is applied. We define $TLOG = u$, the number of times FIX-IT III is applied. A priori, $TLOG = \infty$ is possible, but we shall give conditions that imply $E[TLOG] < \infty$ which in turn implies that the FIX-IT Algorithm will terminate with probability one.

We define a *Moser Tree* to be a finite rooted tree T whose vertices are labelled by $\alpha \in I$. The depth of a node is its distance from the root and the depth of a tree is the maximal depth of its nodes. We require

1. If a node labelled β is a child of a node labelled α then $\beta \sim \alpha$.
2. If two nodes at the same depth have labels β, γ then it *cannot* be that $\beta \sim \gamma$. (In particular, the labels at a given depth are distinct.)

Note, however, that many nodes may, and often will, have the same label. We define $p[T]$ as the product of the $p[\alpha]$ where α ranges over the labels of the nodes. To clarify, when α appears u times the factor $p[\alpha]$ appears u times.

Theorem 5.7.1 *Suppose the sum of $p[T]$ over all Moser Trees T is convergent and let s denote its sum. Then $E[TLOG] \leq s$.*

Let $\alpha_1, \ldots, \alpha_u$ be a prefix of LOG. We associate to it a Moser tree T_u. The root is labelled α_u. Now let t run from $u-1$ down to 1. If we do not have $\alpha_t \sim \alpha_{t'}$ for some $t < t' \leq u$ for which $\alpha_{t'}$ has been already placed in T_u then we ignore α_t. Otherwise, amongst all such t' select one such that the node labelled $\alpha_{t'}$ is at the greatest depth. (In case of ties, select any one such $\alpha_{t'}$.) Add a node with label α_t and make it the child of that node labelled $\alpha_{t'}$.

We claim $T = T_u$ will be a Moser tree. When a node with label β is created as a child of a node with label α we must have $\alpha \sim \beta$. Now suppose at depth D there are two nodes with labels β, γ and $\beta \sim \gamma$. One of them would have been created first, say the one with label $\beta = \alpha_{t'}$. Later in the process we reach $\gamma = \alpha_t$. As $\gamma \sim \beta$ the node with label γ will be placed at depth at least $D+1$, contradicting our assumption.

Remark: The prefix $\alpha_1 \ldots, \alpha_u$ depends on the selection process used in FIX-IT III. In some sense the Moser Tree T_u encapsulates the critical information leading to the choice of α_u. If, say, FIX-IT III had given priority to the nodes in the tree then it would have begun with precisely the nodes in the tree, in any order for which children come before their parent.

Example: Suppose Ω is the English alphabet and $\alpha \sim \beta$ if they are equal or one or two apart in alphabetical order. Consider the prefix $RFSPTR$. The Moser Tree begins (at the end) with root R. T is a child of R. P is a child of R. S is a child of P, F is ignored, and R is a child of S. Had F been given low priority in FIX-IT III the prefix would have been $RSPTR$. Had, further, P been given high priority the prefix would have been $RPSTR$.

Let $u < v$ and assume LOG has prefix $\alpha_1 \cdots \alpha_v$. We claim the Moser Trees T_u, T_v are not equal. Indeed, suppose they are. As they have roots α_u, α_v it must be that $\alpha_u = \alpha_v$. All $1 \leq j \leq u$ with α_j in T_u would also have α_j in T_v and T_v would have the additional node α_v, showing the two trees differ. A LOG of length u will then generate u distinct Moser trees. Hence

$$E[TLOG] = \sum_T \Pr[T = T_n \text{ for some } n]$$

Theorem 5.7.2 *For any Moser Tree T*

$$\Pr[T = T_n \textit{ for some } n] \leq p(T)$$

It is helpful to *preprocess* the random choices of $C[v]$. For all integers $t \geq 0$ let $C[v,t]$ have the distribution of $C[v]$ and let all $C[v,t]$ be mutually independent. At step FIX-IT I, v is given $C[v,0]$. At step FIX-IT IV, if $C[v,t]$ has already been used then v is given $C[v,t+1]$. Given the Moser Tree T say vertex v appears in $A[\alpha_{i_0}], \ldots, A[\alpha_{i_u}]$ where the nodes are listed in order of depth, the highest depth first. Any $s < i_u$ with $v \in A[\alpha_s]$ must appear in the Moser Tree. Hence when FIX-IT IV is applied to $A[\alpha_{i_t}]$ it will use $C[v,t]$. For some $T_n = T$ (regardless of n) it is necessary that $BAD[\alpha_t]$ for all t. The $BAD[\alpha_t]$ have probability $p[\alpha_t]$. Critically, as the "coin flips" $C[v,t]$ are mutually independent and none is used twice the events $BAD[\alpha_t]$ are mutually independent and so the probability that they all hold is the product of the probabilities, namely $p(T)$.

Example: Let $A[\alpha] = \{1, 2, 3\}$, $A[\beta] = \{2, 3, 4\}$ and let T consist of root β with single child α. Let $BAD[\alpha], BAD[\beta]$ be the events that $A[\alpha], A[\beta]$ are monochromatic. In order for $T = T_n$ for some n it is necessary that $C[1, 0]$,$C[2, 0]$,$C[3, 0]$ be the same and that $C[2, 1]$, $C[3, 1]$,$C[4, 0]$ be the same. These are mutually independent events and the probability they both hold is $(1/4)^2$. If $T = T_n$ there cannot be any other γ with 2 or 3 or 4 in $A[\gamma]$ that appears before β, as that γ would have been a node in the Moser tree T_n. There cannot be any other γ with $1 \in A[\gamma]$ that appears before α as that α would also have been a node in the Moser tree T_n.

Calculating the sum of $p[T]$ over all Moser trees can be a daunting task. Instead we find a larger sum. We call a labelled (with I) rooted tree a *weak* Moser tree if the following conditions hold:

1. If a node labelled β is a child of a node labelled α then $\beta \sim \alpha$.

2. The labels of the children of a node are distinct.

Weak Moser trees have a nice recursive structure. For $\alpha \in I$ let $w(\alpha)$ denote the (possibly infinite) sum of $p[T]$ over all weak Moser trees with root labelled α. For $\alpha \in I$ let $w(D, \alpha)$ denote the sum of $p[T]$ over all weak Moser trees with root labelled α and depth at most D. Weak Moser trees with root α and depth at most D decompose into the root α and some (maybe none) Moser trees with roots $\beta \sim \alpha$, all of whom have depth at most $D - 1$. Thus w is given by the recursive system

$$w(D, \alpha) = p(\alpha) \prod_{\beta \sim \alpha} (1 + w(D - 1, \beta)) \tag{5.9}$$

The only tree with root α and depth 0 consists solely of the root and has $p(T) = p(\alpha)$. This yields the initial condition $w(0, \alpha) = p(\alpha)$.

Theorem 5.7.3 *Suppose there exist $x[\alpha] \geq p[\alpha]$ for $\alpha \in I$ such that*

$$x(\alpha) \geq p(\alpha) \prod_{\beta \sim \alpha} (1 + x(\beta)) \tag{5.10}$$

Then $w(\alpha) \leq x(\alpha)$. Further, $E[TLOG] \leq \sum_{\alpha \in I} x(\alpha)$.

Proof: We show $w(D, \alpha) \leq x(\alpha)$ for all $\alpha \in I$ by induction on D. For $D = 0$, $w(0, \alpha) = p(\alpha) \leq x(\alpha)$. Suppose, by induction on D, that $w(D - 1, \beta) \leq x(\beta)$ for all $\beta \in I$. From (5.9,5.10) $w(D, \alpha) \leq x(\alpha)$ for all $\alpha \in I$. Thus $w(\alpha) = \lim_{D \to \infty} w(D, \alpha) \leq x(\alpha)$. As all Moser trees are weak Moser trees, $E[TLOG]$ is at most the sum of $p(T)$ over all weak Moser trees, which is at most $\sum_{\alpha \in I} x(\alpha)$.

The symmetric case is particularly nice, and often occurs in applications. Suppose all $p[\alpha] \leq p$ and for all α, $|\{\beta : \beta \sim \alpha\}| \leq d$. We apply Theorem 5.7.3 with all $x(\alpha) = x$. If there exists $x \geq p$ such that $x \geq p(1 + x)^d$ then $E[TLOG] \leq x|I|$. $x(1 + x)^{-d}$ has maximal value $(d - 1)^{d-1}d^{-d}$, given at $x = (d - 1)^{-1}$.

Theorem 5.7.4 *If $p \leq (d - 1)^{d-1}d^{-d}$ then $E[TLOG] \leq |I|/(d - 1)$.*

Theorem 5.7.5 *If* $epd \leq 1$ *then* $E[TLOG] \leq |I|/(d-1)$.

Moser has also given an alternative analysis of the algorithm based on what he calls an entropy compression argument. The basic idea here is to show that if the algorithm runs without terminating for a long time, then the LOG constructed enables us to compress the random string to a shorter one, and this is impossible. Rather than describing the argument for the general case, we give here only one simple and elegant illustration, given in Grytczuk, Kozik and Micek (2013).

A *repetition of length* h in a sequence is two identical adjacent blocks, each consisting of h consecutive elements. A sequence is *nonrepetitive* if it contains no repetitions. Thus, for example, the sequence 1231241 is nonrepetitive, while 1213413451 is not, as it contains the repetition 134134. Thue proved in 1906 that there is an infinite nonrepetitive sequence over an alphabet of 3 symbols. An extension is proved in Alon, Grytczuk, Hałuszczak and Riordan (2002) using the Local Lemma: the vertices of any graph with maximum degree d can be colored by $O(d^2)$ colors so that every path in the graph is nonrepetitive. A variant in which the allowed colors for each vertex must lie in a list associated with the vertex has been considered as well. Here we prove the following.

Theorem 5.7.6 (Grytczuk et al. (2013)) *For every* $n \geq 1$ *and every sequence of lists of symbols* $L_1, L_2, \ldots, L_n$*, each of size* 5*, there is a nonrepetitive sequence* $s_1, s_2, \ldots, s_n$*, where* $s_i \in L_i$*. Moreover, there is a randomized algorithm that finds such a sequence in expected time polynomial in* n*, given the lists.*

Note that by König's Lemma the above implies that any infinite sequence of lists of size 5 admits a nonrepetitive sequence of symbols chosen from the lists. We note also that it has been conjectured that lists of size 3 suffice, see the paragraph following the proof for more about lists of size 3 and 4.

The proof of the theorem is by a simple algorithm: the sequence is generated by choosing symbols randomly, independently and uniformly from the lists, where every time a repetition occurs, the repeated block is erased and the process continues. More formally, consider the following algorithm for generating a nonrepetitive sequence $s_1, s_2, \ldots, s_n$ from the lists $L_1, L_2, \ldots, L_s$, each of size 5.

Starting with $i = 1$, as long as $i \leq n$ perform an iteration as follows: let s_i be a random element of L_i. If the sequence $s_1, s_2, \ldots, s_i$ is nonrepetitive, increase i to $i + 1$ and go to the next iteration. Otherwise, note that, crucially, there is a unique repetition ending at s_i, let it be $s_{i-2h+1}, \ldots, s_{i-h}, s_{i-h+1}, \ldots, s_i$. In this case replace i by $i - h + 1$ and proceed to the next iteration. (This includes the case $h = 1$ where $s_i = s_{i-1}$ and s_i is deleted.)

Note that if the algorithm terminates, it generates a nonrepetitive sequence of length n, as required. To complete the proof we show that with high probability it terminates after $O(n)$ iterations. To estimate the probability it does not terminate in M iterations, fix an arbitrary order of the symbols in each list and let let $r_j \in [5]$ be the position of the chosen element in iteration j ($1 \leq j \leq M$). Define a sequence $d_1, \ldots, d_m$ where $d_1 = 1$ and d_j is the difference between the value of i at the end of iteration j and its value at the end of iteration $j - 1$. Thus $d_j = 1$ if in iteration

number j no repetition is obtained, otherwise it is $-h+1$ where h is the length of the repeated block obtained. The LOG corresponding to the run of the algorithm is (D_M, S_M), where $D_M = (d_1, \ldots, d_M)$ and $S_M = (s_1, s_2 \ldots, s_\ell)$ is the sequence obtained after M iterations. The crucial fact is the following:

Fact: Every LOG which can be obtained corresponds to exactly one sequence $r_1, r_2, \ldots, r_M$. Therefore the probability to get it is 5^{-M}.

To prove the fact it suffices to show that knowing the LOG (D_M, S_M) we can reconstruct r_M and (D_{M-1}, S_{M-1}), where $D_{M-1} = (d_1, \ldots, d_{M-1})$ and S_{M-1} is the sequence the algorithm generates after $M-1$ iterations. The same process will then enable us to reconstruct, by induction, $r_{M-1}, \ldots, r_1$.

Knowing D_M clearly reveals D_{M-1}. If $d_M = 1$ then r_M is just the position of s_ℓ in the list L_ℓ, and S_{M-1} is S_M without its last symbol. Otherwise $d_M = -h+1$ where h is the length of the last repeated block. In this case the h symbols that have been erased from S_{M-1} together with the last chosen symbol to get S_M are equal to the last h symbols of S_M, in order, and thus in this case $S_{M-1} = s_1, \ldots, s_\ell, s_{\ell-h+1}, \ldots, s_{\ell-1}$ and r_M is the location of s_ℓ in the list $L_{\ell+h}$. This proves the fact.

Each possible sequence $D_M = (d_1, \ldots, d_m)$ can be encoded by a sequence over $+, -$ as follows: for each j, $1 \leq j \leq M$, in order, write $+$ once, followed by $|d_j - 1|$ times $-$ (note that $d_j - 1$ is zero when $d_j = 1$). Thus the total number of $+$ in the sequence is M, and the the total number of $-$ is exactly the number of symbols discarded during the process, which is at most $M-1$ (and at least $M-n$, but we will not use this last fact here). This shows that the number of possibilities for D_M in the LOG is smaller than 2^{2M} and hence the number of possibilities for the LOG (D_M, S_M) is at most $2^{2M}5^n$. By the fact this implies that the probability the algorithm does not terminate after M iterations is at most $4^M 5^{n-M}$, which is tiny for, say, $M = 8n$. Therefore, the algorithm terminates, with high probability, after less than $8n$ iterations and the expected number of iterations until termination is $O(n)$. This completes the proof of the theorem.

The randomized algorithm described above is well defined for lists of any size. It has been conjectured that for some $M = M(n)$ the algorithm terminates with high probability when all lists have size 3. We give this as an Exercise when the lists all have size 4. Extensive simulation (at least, in the case where all L_i are the same) indicates that one can take $M = O(n)$ even if each list is of size 3, but this remains a conjecture.

5.8 EXERCISES

1. (*) Prove that for every integer $d > 1$ there is a finite $c(d)$ such that the edges of any bipartite graph with maximum degree d in which every cycle has at least $c(d)$ edges can be colored by $d+1$ colors so that there are no two adjacent edges with the same color and there is no two-colored cycle.