Randomness and Computation
or, “Randomized Algorithms”

Mary Cryan
School of Informatics
University of Edinburgh

Examples of SAT, k-SAT
Example of a SAT question:
\[(x_1 \lor x_8 \lor \overline{x}_6) \land (\overline{x}_4 \lor \overline{x}_7) \land (x_5 \lor x_7 \lor x_4 \lor x_2).\]

▶ For the formula above, easy to see there is a (many) satisfying assignment(s) to the \(x_i\) variables (any with \(x_1 = 1, x_4 = 0, x_2 = 1\) would do, for example).

▶ In general, the SAT problem is NP-complete (we believe there is no polynomial-time algorithm).

Example of a 2-SAT question:
\[(x_1 \lor \overline{x}_2) \land (\overline{x}_1 \lor x_3) \land (x_1 \lor x_2) \land (x_4 \lor x_3) \land (x_4 \lor \overline{x}_1).\]

▶ There is a polynomial-time algorithm (either randomized, as we see today, or deterministic) to solve 2-SAT.

▶ The 3-SAT problem, and k-SAT for all \(k > 3\), are all NP-complete.

2-SAT Randomized Algorithm

We will design a simple randomized algorithm for 2-SAT, and analyse its performance by analogy to a Markov chain.

Algorithm 2SATRandom(\(n; C_1 \land C_2 \land \ldots \land C_t\))

1. Assign arbitrary values to each of the \(x_i\) variables.
2. \(t \leftarrow 0\)
3. while \((t < 2mn^2 \text{ and some clause is unsatisfied})\) do
   4. Choose an arbitrary \(C_h\) from all unsatisfied clauses;
   5. Choose one of the 2 literals in \(C_h\) uniformly at random and flip the value of its variable;
6. if (we end with a satisfying assignment) then
   7. return this assignment to the \(x_1, \ldots, x_n\) else
   8. return FAILED.

Note that arbitrary is very different from random.
2-SAT Randomized Algorithm

Imagine Algorithm 2SATRANDOM running on our 2SAT example, with the initial assignment being \( x_i = 0 \) for all \( i \in [n] \).

\[
( x_1 \lor \bar{x}_2 ) \land ( \bar{x}_1 \lor x_3 ) \land ( x_1 \lor x_2 ) \land ( x_4 \lor \bar{x}_3 ) \land ( x_4 \lor x_1 ).
\]

- Then \( ( x_1 \lor x_2 ) \) is the sole unsatisfied clause.
- Flipping the value of \( x_2 \) (say) from 0 to 1, will ensure that \( ( x_1 \lor x_2 ) \) now becomes satisfied.
- However, making this flip would also change the assignment for \( ( x_1 \lor x_2 ) \), making this clause now unsatisfied.
- This is a balanced consequence overall (number of satisfied clauses stays the same). Note that a similar scenario would arise had we instead flipped \( x_1 \) to satisfy \( ( x_1 \lor x_2 ) \) (we would have violated \( ( x_4 \lor \bar{x}_3 ) \) in that case).

*However, there are examples where a flip might end up violating many clauses.* So it’s not so helpful for us to use “number of clauses satisfied” as our measure of progress.

---

2-SAT Randomized Algorithm - Analysis

Consider an (unknown so far) satisfying assignment \( S \in \{0, 1\}^n \) that makes our 2SAT formula \( \phi \) true (satisfies all the clauses).

Our “measure of progress” will be the number of indices \( k \) such that \( x_k = S_k \), \( (x_1, \ldots, x_n) \) being the current assignment.

We will analyse the expected number of steps before \( (x_1, \ldots, x_n) \) becomes \( S \).

- This of course assumes the formula \( \phi \) has some satisfying assignment.
- Note that if \( \phi \) does not have any satisfying assignment, Algorithm 2SATRANDOM always returns FAILED (as it should).

---

2-SAT Randomized Algorithm - Analysis

To analyse the behaviour of Algorithm 2SATRANDOM when given a 2CNF formula \( \phi \) that is satisfiable, we need some definitions.

**Definition**

For our given satisfiable 2SAT formula \( \phi \), let \( S \) be some satisfying assignment for \( \phi \).

Let \( (x_1', \ldots, x_n') \) denote the assignment to the logical variables after the \( i \)th iteration of the loop at 3.

Let \( X_t \) denote the number of variables of the assignment \( (x_1', \ldots, x_n') \) having the same value as in \( S \).

We work with the \( X_t \) variable mainly, and bound the time before it reaches the value \( n \).

Some observations:

- If \( X_t \) ever hits the 0, and \( \phi \) is unsatisfied, we are guaranteed that at the next step, \( X_{t+1} = 1 \).

\[
\Pr[X_{t+1} = 1 \mid ((X_t = 0) \& \phi \text{ unsat})] = 1.
\]

- Alternatively, suppose \( X_t = j \) for some value \( j \in [1, \ldots, n-1] \) and that \( \phi \) is unsatisfied.

Then on any of the individual unsatisfied clauses, we know the current assignment \( x' \) must differ from \( S \) on at least one of the two variables. Hence with probability at least \( 1/2 \), we will increase the value of \( X_t \) by 1 (and with probability at most \( 1/2 \) decrease the value of \( X_t \) by 1/2).

\[
\Pr[X_{t+1} = j + 1 \mid ((X_t = j) \& \phi \text{ unsat})] \geq 1/2; \quad \Pr[X_{t+1} = j - 1 \mid ((X_t = j) \& \phi \text{ unsat})] \leq 1/2.
\]

---

RC (2017/18) – Lecture 14 – slide 5

RC (2017/18) – Lecture 14 – slide 6

RC (2017/18) – Lecture 14 – slide 7

RC (2017/18) – Lecture 14 – slide 8
2-SAT Randomized Algorithm - Analysis

We want to imagine the progress of 2SAT RANDOM as a Markov chain on the states 0, 1, ..., n. Our concern is bounding the expected number of steps t for \( X_t \) to hit the state n (from an arbitrary starting point).

- Markov chains should be memoryless, and this is problematic.
- The value for \( \Pr[X_{t+1} = j + 1 \mid (X_t = j) \& \phi \text{ unsat}] \) can be 1/2 or 1 depending on how many variables of the chosen clause currently disagree with \( S \). This may have been affected by earlier flips done by the algorithm.
- We choose to “tweak” the probabilities and study the process on \( \{0, 1, 2, ..., n\} \) defined by the variable \( Y_t \) on the next slide.

For any \( j = 0, ..., n - 1 \), define \( h_j \) to be the expected number of steps to hit n starting from \( j \).

We have \( h_n = 0 \) and \( h_0 = h_1 + 1 \) for the “end cases”.

We will use \( Z_j \), for 0, 1, ..., \( n - 1 \), to be the random variable for the number of steps \( t \) for \( X_t \) to hit n. Our concern is bounding the expected number of steps for \( 2\text{SAT RANDOM} \) to find a satisfying assignment is at most \( \max_j h_j \) (may well be better).

We will bound \( h_j \) for every \( j = 0, 1, ..., n \).

The Markov chain \( Y_t \)

Consider the Markov chain \( Y_0, Y_1, ..., Y_t, ... \) such that

\[
Y_0 = X_0; \\
\Pr[Y_{t+1} = 1 \mid (Y_t = 0) \& \phi \text{ unsat}] = 1; \\
\Pr[Y_{t+1} = j + 1 \mid (Y_t = j) \& \phi \text{ unsat}] = 1/2; \\
\Pr[Y_{t+1} = j - 1 \mid (Y_t = j) \& \phi \text{ unsat}] = 1/2.
\]

Clearly, the expected number of steps for \( X_t \) to hit n is \( \leq \) that for \( Y_t \).
2-SAT Randomized Algorithm - Analysis

We show by induction that for $j = 0, \ldots, n - 1$,

$$h_j = h_{j+1} + 2j + 1.$$

Proof.

Base case: If $j = 0$, $2j + 1 = 1$, and we were given $h_0 = h_1 + 1$.

Inductive step: Suppose this was true for $j = k - 1$ (we had $h_{k-1} = h_k + 2(k - 1) + 1$, this is our (IH)). Now consider $j = k$.

By the “middle case” of our system of equations,

$$h_k = \frac{h_{k-1} + h_{k+1}}{2} + 1$$

$$= \frac{h_k + 2(k - 1) + 1}{2} + \frac{h_{k+1}}{2} + 1 \quad \text{by our (IH)}$$

$$= \frac{h_k}{2} + \frac{h_{k+1}}{2} + \frac{2k + 1}{2}$$

Subtracting $\frac{h_k}{2}$ from each side, this is equivalent to

$$h_k = h_{k+1} + 2k + 1,$$

as claimed.

2-SAT Randomized Algorithm - Analysis

Lemma (Lemma 7.1)

Assume that the given 2CNF formula has a satisfying assignment, and that 2SATRANDOM is allowed to carry out as many iterations as it wants to find a satisfying assignment. Then the expected number of iterations of 3, to find that assignment at most $n^2$.

Proof.

We showed that the expected number of iterations is at most $\max_{j=0,\ldots,n-1}(h_j)$. We now know the max is $h_0$. Applying $h_k = h_{k+1} + 2k + 1$ iteratively, we have

$$h_0 = \sum_{k=0}^{n-1} (2k + 1) + h_n$$

$$= 2 \sum_{k=0}^{n-1} k + n + 0$$

$$= 2 \frac{(n-1)n}{2} + n = n^2.$$

Probability of failure

Theorem

Algorithm 2SATRANDOM is parametrized by $m$, and the algorithm will perform up to $mn^2$ iterations of the loop.

Then, when there is a satisfying assignment for $\phi$, the probability that 2SATRANDOM does not discover one, is at most $2^{-m}$.

Proof.

Markov’s Inequality.

Reading and Doing

Reading

▶ This material is from Section 7.1 of [MU].

▶ Section 7.4 from the book is interesting (we were looking at a random walk on the line today).

Doing

▶ Exercises 7.3 and 7.7 from [MU].