6.4. Sample and Modify

- Thus far we have constructed random structures with the desired properties directly
- In some cases it is easier to work indirectly, breaking the argument into two stages
- First we construct a random structure that does not have the required properties
- In the second stage we then modify the random structure so that it does have the required property

6.4.1. Application: Independent Sets

- An **independent set** (IS) in a graph \( G \) is a set of vertices with no edges between them
- Finding the largest IS in a graph is an NP-hard problem
- The probabilistic method can yield bounds on the size of the largest IS of a graph:

**Theorem 6.5:** Let \( G = (V, E) \) be a graph on \( n \) vertices with \( m \) edges. Then \( G \) has an IS with at least \( n^2 / 4m \) vertices.
**Proof:** Let \( d = 2 \frac{m}{n} \) be the average degree of the vertices in \( G \). Consider the following randomized algorithm:

1. Delete each vertex of \( G \) (and its incident edges) independently with probability \( 1 - \frac{1}{d} \).
2. For each remaining edge, remove it and one of its adjacent vertices.

The remaining vertices form an IS, since all edges have been removed.

This is an example of the sample-and-modify technique. We first sample the vertices, and then we modify the remaining graph.

Let \( X \) be the number of vertices that survive the first step of the algorithm. Since the graph has \( n \) vertices and since each vertex survives with probability \( \frac{1}{d} \), it follows that
\[
E[X] = \frac{n}{d}.
\]

Let \( Y \) be the number of edges that survive the first step. There are \( m = n \left( \frac{2m}{n} \right)/2 = \frac{nd}{2} \) edges in the graph, and an edge survives if and only if its two adjacent vertices survive. Thus
\[
E[Y] = \frac{nd}{2} \left( \frac{1}{d} \right)^2 = \frac{n}{2d}.
\]
The second step of the algorithm removes all the remaining edges and at most $Y$ vertices. When the algorithm terminates, it outputs an IS of size at least $X - Y$, and

$$E[X - Y] = \frac{n}{d} - \frac{n}{2d} = \frac{n}{2d}.$$ 

The expected size of the IS generated by the algorithm is $n/2d$, so the graph has an IS with at least $n/2d = n/2 (2m/n) = n^2/4m$ vertices.

6.4.2. Application: Graphs with Large Girth

- Consider the *girth* of a graph, which is the length of its smallest cycle
- Intuitively we expect dense graphs to have small girth
- We can show, however, that there are dense graphs with relatively large girth

**Theorem 6.6:** For any integer $k \geq 3$ there is a graph with $n$ nodes, at least $\frac{1}{4n^{1+1/k}}$ edges, and girth at least $k$. 
**Proof:** We first sample a random graph $G \in G_{n,p}$ with $p = n^{1/k-1}$. Let $X$ be the number of edges in the graph. Then

$$E[X] = p \binom{n}{2} = \frac{n(n-1)}{2n} n^{1/k} = \frac{1}{2} \left( 1 - \frac{1}{n} \right) n^{1/k+1}.$$  

Let $Y$ be the number of cycles in the graph of length at most $k-1$. Any specific possible cycle of length $i$, where $3 \leq i \leq k-1$, occurs with probability $p^i$. Also, there are $\binom{n}{i}(i-1)!$ possible cycles of length $i$: first choose the $i$ vertices, then consider the possible orders, and keep in mind that reversing the order yields the same cycle.

$$E[Y] = \sum_{i=3}^{k-1} \binom{n}{i} \frac{(i-1)!}{2} p^i \leq \sum_{i=3}^{k-1} n^i p^i = \sum_{i=3}^{k-1} n^i / k < kn^{(k-1)/k}.$$  

We modify the original randomly chosen graph $G$ by eliminating one edge from each cycle of length up to $k-1$. The modified graph therefore has girth at least $k$.

When $n$ is sufficiently large, the expected number of edges in the resulting graph is

$$E[X - Y] \geq \frac{1}{2} \left( 1 - \frac{1}{n} \right) n^{1/k+1} - kn^{(k-1)/k} \geq \frac{1}{4} n^{1/k+1}.$$  

Hence there exists a graph with at least $\frac{1}{4} n^{1+1/k}$ edges and girth at least $k$.  

$\blacksquare$
6.5. The Second Moment Method

- The second moment method typically makes use of the following inequality, which is easily derived from Chebyshev's inequality

**Theorem 6.7:** If $X$ is a nonnegative integer-valued RV, then

$$
\Pr(X = 0) \leq \frac{\text{Var}[X]}{(\mathbb{E}[X])^2}.
$$

**Proof:**

$$
\Pr(X = 0) \leq \Pr(|X - \mathbb{E}[X]| \geq \mathbb{E}[X]) \leq \frac{\text{Var}[X]}{(\mathbb{E}[X])^2}.
$$

6.5.1. Threshold Behavior in Random Graphs

- The second moment method can be used to prove the threshold behavior of certain random graph properties

- i.e., in the $G_{n,p}$ model it is often the case that there is a threshold function $f$ such that:

  a) when $p$ is just less than $f(n)$, almost no graph has the desired property; whereas

  b) when $p$ is just larger than $f(n)$, almost every graph has the desired property
**Theorem 6.8:** In $G_{n,p}$, suppose that $p = f(n)$, where $f(n) = o(n^{-2/3})$. Then, for any $\varepsilon > 0$ and for sufficiently large $n$, the probability that a random graph chosen from $G_{n,p}$ has a clique of four or more vertices is less than $\varepsilon$.

Similarly, if $f(n) = \omega(n^{-2/3})$ then, for sufficiently large $n$, the probability that a random graph chosen from $G_{n,p}$ does not have a clique with four or more vertices is less than $\varepsilon$.

**Proof:** Consider first the case in which $p = f(n)$ and $f(n) = o(n^{-2/3})$. $C_1, \ldots, C_{\binom{n}{4}}$ is an enumeration of all the subsets of four vertices in $G$. Let $X_i = \begin{cases} 1 & \text{if } C_i \text{ is a 4-clique} \\ 0 & \text{otherwise} \end{cases}$

- Let

$$X = \sum_{i=1}^{\binom{n}{4}} X_i,$$

- so that

$$\mathbb{E}[X] = \binom{n}{4} p^6 = \Theta(n^4 p^6) = \Theta\left((pn^{2/3})^6\right) = o(1).$$
Now $\mathbf{E}[X] = o(1)$, which means that $\mathbf{E}[X] < \epsilon$ for sufficiently large $n$. Since $X$ is a nonnegative integer-valued RV, it follows that $\Pr(X \geq 1) \leq \mathbf{E}[X] < \epsilon$. Hence, the probability that a random graph chosen from $G_{n,p}$ has a clique of four or more vertices is less than $\epsilon$.

Consider then the case when $p = f(n)$ and $f(n) = \omega(n^{-2/3})$. In this case, $\mathbf{E}[X] \to \infty$ as $n$ grows large. This is not sufficient to conclude that, w.h.p., a graph chosen random from $G_{n,p}$ has a clique of at least four vertices. We can, however, use Theorem 6.7 to prove that $\Pr(X = 0) = o(1)$ in this case.

To do so we must show that $\mathbf{Var}[X] = o((\mathbf{E}[X])^2)$. We begin with the following useful formula.

**Lemma 6.9:** Let $Y_i$, $i = 1, \ldots, m$, be 0-1 RVs, and let $Y = \sum_{i=1}^{m} Y_i$. Then

$$\mathbf{Var}[Y] \leq \mathbf{E}[Y] + \sum_{1 \leq i, j \leq m \atop i \neq j} \mathbf{Cov}(Y_i, Y_j)$$

We wish to compute

$$\mathbf{Var}[X] = \mathbf{Var} \left[ \sum_{i=1}^{\binom{n}{4}} X_i \right].$$
To apply (6.9), we see that we need to consider the covariance of the $X_i$. If $|C_i \cap C_j| = 0$ then the corresponding cliques are disjoint, and it follows that $X_i$ and $X_j$ are independent. Hence, $E[X_i X_j] - E[X_i]E[X_j] = 0$. The same holds if $|C_i \cap C_j| = 1$.

If $|C_i \cap C_j| = 2$, then the cliques share one edge. For both cliques to be in the graph, the eleven corresponding edges must appear in the graph. Hence, now $E[X_i X_j] - E[X_i]E[X_j] \leq E[X_i X_j] \leq p^{11}$. There are $\binom{n}{6}$ ways to choose the six vertices and $\binom{6}{2;2;2}$ ways to split them into $C_i$ and $C_j$.

If $|C_i \cap C_j| = 3$, then the cliques share three edges. For both cliques to be in the graph, the nine corresponding edges must appear in the graph. Hence, $E[X_i X_j] - E[X_i]E[X_j] \leq E[X_i X_j] \leq p^9$. There are $\binom{n}{5}$ ways to choose the five vertices, and $\binom{5}{3;1;1}$ ways to split them into $C_i$ and $C_j$.

Recall that $E[X] = \binom{n}{4}p^6 = \Theta(n^4p^6)$, hence $(E[X])^2 = \Theta(n^8p^{12})$ and $p = f(n) = \omega(n^{-2/3})$.

$$\text{Var}[X] \leq \binom{n}{4}p^6 + \binom{n}{6}\binom{6}{2;2;2}p^{11} + \binom{n}{5}\binom{5}{3;1;1}p^9 = o(n^8p^{12})$$
$o(n^8p^{12}) = o((E[X])^2)$, since

$$(E[X])^2 = \left(\binom{n}{4}p^6\right)^2 = \Theta(n^8p^{12}).$$

Theorem 6.7 now applies, showing that $\Pr(X = 0) = o(1)$ and thus the second part of the theorem.

E.g.,

$$\frac{n^4p^6}{n^8p^{12}} = \frac{1}{n^4p^6} = o\left(\frac{1}{n^4(n^{-2/3})^6}\right) = o(1).$$

6.7. The Lovasz Local Lemma

- Let $E_1, \ldots, E_n$ be a set of bad events in some probability space
- We want to show that there is an element in the sample space not included in any $E_i$
- This would be easy if the events were mutually independent; i.e., iff, for any subset $I \subseteq [1, n]$

$$\Pr\left(\bigcap_{i \in I} E_i\right) = \prod_{i \in I} \Pr(E_i)$$
• Also, if \( E_1, \ldots, E_n \) are mutually independent then so are \( \bar{E}_1, \ldots, \bar{E}_n \).

• If \( \Pr(E_i) < 1 \) for all \( i \), then

\[
\Pr\left( \bigcap_{i \in I} \bar{E}_i \right) = \prod_{i=1}^{n} \Pr(\bar{E}_i) > 0
\]

and there is an element of the sample space that is not included in any bad event.

• Mutual independence is too much to ask for.

• The Lovasz local lemma generalizes the preceding argument to the case where the \( n \) events are not mutually independent but the dependency is limited.

An event \( E \) is mutually independent of the events \( E_1, \ldots, E_n \) if, for any subset \( I \subseteq [1, n] \),

\[
\Pr\left( E \bigg\| \bigcap_{j \in I} E_j \right) = \Pr(E)
\]

The dependency between events can be represented in terms of a dependency graph.

**Definition 6.1:** A dependency graph for a set of events \( E_1, \ldots, E_n \) is a graph \( G = (V, E) \) such that \( V = \{1, \ldots, n\} \) and, for \( i = 1, \ldots, n \), event \( E_i \) is mutually independent of the events \( \{E_j \mid (i, j) \notin E\} \).
The symmetric version of the Lovasz local lemma:

**Theorem 6.11 [Lovasz Local Lemma]:**

Let \( E_1, \ldots, E_n \) be a set of events, and assume that the following hold:

1. for all \( i \), \( \Pr(E_i) \leq p \);
2. the degree of the dependency graph given by \( E_1, \ldots, E_n \) is bounded by \( d \);
3. \( 4dp \leq 1 \).

Then

\[
\Pr\left(\bigcap_{i=1}^{n} \overline{E_i}\right) > 0.
\]

**6.7.1. Application: Edge-Disjoint Paths**

- Assume that \( n \) pairs of users need to communicate using edge-disjoint paths on a given network.
- Each pair \( i = 1, \ldots, n \) can choose a path from a collection \( F_i \) of \( m \) paths.
- We show using the Lovasz local lemma that, if the possible paths do not share too many edges, then there is a way to choose \( n \) edge-disjoint paths connecting the \( n \) pairs.
Theorem 6.12: If any path in $F_i$ shares edges with no more than $k$ paths in $F_j$, where $i \neq j$ and $8nk/m \leq 1$, then there is a way to choose $n$ edge-disjoint paths connecting the $n$ pairs.

Proof: Consider the probability space defined by each pair choosing a path $I+U@R$ from its set of $m$ paths. Define $E_{i,j}$ to represent the event that the paths chosen by pairs $i$ and $j$ share at least one edge. Since a path in $F_i$ shares edges with no more than $k$ paths in $F_j$,

$$p = \Pr(E_{i,j}) \leq \frac{k}{m}.$$ 

Let $d$ be the degree of the dependency graph. Since event $E_{i,j}$ is independent of all events $E_{i',j'}$ when $i' \notin \{i, j\}$ and $j' \notin \{i, j\}$, we have $d < 2n$. Since

$$4dp < \frac{8nk}{m} \leq 1,$$

all of the conditions of the Lovasz local lemma are satisfied, proving

$$\Pr\left(\bigcap_{i \neq j} \overline{E}_{i,j}\right) > 0.$$ 

- Hence, there is a choice of paths such that the $n$ paths are edge disjoint. □
6.7.2. Application: Satisfiability

For the $k$-satisfiability ($k$-SAT) problem, the formula is restricted so that each clause has exactly $k$ literals.

Again, we assume that no clause contains both a literal and its negation, as these clauses are trivial.

We prove that any $k$-SAT formula in which no variable appears in too many clauses has a satisfying assignment.

**Theorem 6.13:** If no variable in a $k$-SAT formula appears in more than $T = 2^k / 4k$ clauses, then the formula has a satisfying assignment.

**Proof:** Consider the probability space defined by giving a random assignment to the variables. For $i = 1, ..., m$, let $E_i$ denote the event that the $i$th clause is not satisfied by the random assignment. Since each clause has $k$ literals, 

$$\Pr(E_i) = 2^{-k}.$$
• $E_i$ is mutually independent of all of those related to clauses that do not share variables with clause $i$

• Each of the $k$ variables in clause $i$ appears in at most $T = 2^k/4k$ clauses, the degree of the dependency graph is $\leq d \leq kT \leq 2^{k-2}$.

• In this case, $4dp \leq 4 \cdot 2^{k-2} 2^{k} \leq 1$, so we can apply the Lovasz local lemma to conclude that

$$\Pr\left(\bigcap_{i=1}^{m} \bar{E}_i\right) > 0$$

hence there is a satisfying assignment for the formula.

7 Markov Chains and Random Walks

• A **stochastic process** $X = \{X(t) : t \in T\}$ is a collection of random variables

• The index $t$ often represents time, and in that case the process $X$ models the value of a random variable $X$ that changes over time

• We call $X(t)$ the **state** of the process at time $t$

• We use $X_t$ interchangeably with $X(t)$

• If, for all $t$, $X_t$ assumes values from a countably infinite set, then $X$ is a **discrete space process**
• If $X_t$ assumes values from a finite set then the process is **finite**
• If $T$ is a countably infinite set we say that $X$ is a **discrete time process**
• A special type of discrete time and space stochastic process $X_0, X_1, X_2, \ldots$, in which the value of $X_t$ depends on the value of $X_{t-1}$, but not on the sequence of states that led the system to that value

**Definition 7.1:** A discrete time stochastic process $X_0, X_1, X_2, \ldots$ is a **Markov chain** (MC) if

\[
\Pr(X_t = a_t | X_{t-1} = a_{t-1}, \ldots, X_0 = a_0) = \Pr(X_t = a_t | X_{t-1} = a_{t-1}) = P_{a_{t-1}a_t}.
\]

• The state $X_t$ depends on the previous state $X_{t-1}$ but is independent of the particular history of how the process arrived at state $X_{t-1}$
• This is called the **Markov property** or **memoryless property**
• Markov property does not imply that $X_t$ is independent of the RVs $X_0, X_1, \ldots, X_{t-2}$; any dependency of $X_t$ on the past is captured in the value of $X_{t-1}$
• We can assume that the discrete state space of the MC is $\{0, 1, 2, \ldots, n\}$ (or $\{0, 1, 2, \ldots\}$ if it is countably infinite)
The transition probability
\[ P_{i,j} = \Pr(X_t = j | X_{t-1} = i) \]
of moving from \( i \) to \( j \) in one step

The MC is uniquely defined by the one-step transition matrix:
\[
P = \begin{pmatrix}
P_{0,0} & P_{0,1} & \cdots & P_{0,j} & \cdots \\
P_{1,0} & P_{1,1} & \cdots & P_{1,j} & \cdots \\
\vdots & \vdots & \ddots & \vdots & \ddots \\
P_{i,0} & P_{i,1} & \cdots & P_{i,j} & \cdots \\
\end{pmatrix}
\]

For all \( i, \sum_{j \geq 0} P_{i,j} = 1 \)

Transition matrix representation of a MC is convenient for computing the distribution of future states of the process

Let \( p_i(t) \) denote the probability that the process is at state \( i \) at time \( t \)

Vector \( \vec{p}(t) = (p_0(t), p_1(t), p_2(t), \ldots) \) gives the distribution of the state of the chain at time \( t \)

Summing over all possible states at time \( t-1 \), we have
\[
p_i(t) = \sum_{j \geq 0} p_j(t-1)P_{j,i} \quad \text{or} \quad \vec{p}(t) = \vec{p}(t-1)\mathbf{P}
\]
We represent the probability distribution as a row vector and multiply $\vec{\rho} \mathbf{P}$ instead of $\mathbf{P} \vec{\rho}$ to conform with the interpretation that starting with a distribution $\vec{\rho}(t-1)$ and applying the operand $\mathbf{P}$, we arrive at the distribution $\vec{\rho}(t)$.

For any $m \geq 0$, we define the $m$-step transition probability

$$P_{i,j}^m = \Pr(X_{t+m} = j | X_t = i)$$

as the probability that the chain moves from state $i$ to state $j$ in exactly $m$ steps.

Conditioning on the first transition from $i$, we have

$$P_{i,j}^m = \sum_{k \geq 0} P_{i,k} P_{k,j}^{m-1}$$

Let $\mathbf{P}^{(m)}$ be the matrix whose entries are the $m$-step transition probabilities, so that the entry in the $i$th row and $j$th column is $P_{i,j}^m$.

Applying the above Eqn. yields

$$\mathbf{P}^{(m)} = \mathbf{P} \cdot \mathbf{P}^{(m-1)}$$

by induction on $m$, $\mathbf{P}^{(m)} = \mathbf{P}^m$.

Thus, for any $t \geq 0$ and $m \geq 1$,

$$\vec{\rho}(t + m) = \vec{\rho}(t) \mathbf{P}^m$$
Another useful representation of a MC is by a directed, weighted graph \( D = (V, E, w) \).

- The set of vertices of the graph is the set of states of the chain.
- There is a directed edge \((i, j) \in E \) iff \( P_{i,j} > 0 \), in which case the weight \( w(i, j) = P_{i,j} \).
- Self-loops are allowed.
- For each \( i \) we require that \( \sum_{j:(i,j) \in E} w(i, j) = 1 \).
- A sequence of states visited by the process is represented by a directed path on the graph.
- The probability that the process follows this path is the product of the weights of the path's edges.
Let us consider the probability of going from state 0 to state 3 in exactly three steps.

There are only four such paths: 0-1-0-3, 0-1-3-3, 0-3-1-3, and 0-3-3-3.

The probabilities that the process follows each of these paths are $\frac{3}{32}$, $\frac{1}{96}$, $\frac{1}{16}$, and $\frac{3}{64}$, respectively.

Summing these probabilities, we find that the total probability is $\frac{41}{192}$.

Alternatively, we can simply compute:

$$
\mathbf{P}^3 = \begin{bmatrix}
\frac{3}{16} & \frac{7}{48} & \frac{29}{64} & \frac{41}{192} \\
\frac{5}{48} & \frac{5}{24} & \frac{79}{144} & \frac{5}{36} \\
0 & 0 & 1 & 0 \\
\frac{1}{16} & \frac{13}{96} & \frac{107}{192} & \frac{47}{192}
\end{bmatrix}
$$

Entry $P_{0,3}^3 = \frac{41}{192}$ gives the correct answer.

We want to know the probability of ending in state 3 after three steps when we begin in a state chosen U@R from the four states

$$(\frac{1}{4}, \frac{1}{4}, \frac{1}{4}, \frac{1}{4}) \mathbf{P}^3 = (\frac{17}{192}, \frac{47}{384}, \frac{737}{1152}, \frac{43}{288})$$

The last entry, $\frac{43}{288}$, is the required answer.
7.3. Stationary Distributions

- Recall that if \( P \) is the one-step transition probability matrix of a MC and if \( \tilde{p}(t) \) is the probability distribution of the state of the chain at time \( t \), then

\[
\tilde{p}(t + 1) = \tilde{p}(t)P
\]

- Of particular interest are state probability distributions that do not change after a transition.

**Definition 7.8:** A stationary distribution (also called an equilibrium distribution) of a MC is a probability distribution \( \pi \) such that \( \pi = \pi P \).

A Randomized Algorithm for 2-Satisfiability

- 2-SAT is solvable in polynomial time.
- An input for 2-SAT has exactly two literals per clause.
- An instance of 2-SAT:

\[
(x_1 \lor \overline{x_2}) \land (\overline{x_1} \lor \overline{x_2}) \land (x_1 \lor x_2) \land (x_4 \lor \overline{x_3}) \land (x_4 \lor \overline{x_1})
\]

- One natural approach is to start with an assignment, look for a clause that is not satisfied, and change the assignment so that the clause becomes satisfied.
If there are two literals in the clause, then there are two possible changes to the assignment that will satisfy the clause.

The 2-SAT algorithm decides which of these changes to try randomly.

In the algorithm, $n$ denotes the number of variables in the formula and $m$ is an integer parameter that determines the probability that the algorithm terminates with a correct answer.

2-SAT ALGORITHM:

1. Start with an arbitrary truth assignment.
2. Repeat up to $2mn^2$ times, terminating if all clauses are satisfied:
   a) Choose an arbitrary clause that is not satisfied.
   b) Choose uniformly at random one of the literals in the clause and switch the value of its variable.
3. If a valid truth assignment has been found, return it.
4. Otherwise, return that the formula is unsatisfiable.
If we begin with all variables set to False then the clause $(x_1 \lor x_2)$ is not satisfied. The algorithm might therefore choose this clause and then select $x_1$ to be set to True. In this case the clause $(x_4 \lor \overline{x_1})$ would be unsatisfied and the algorithm might switch the value of a variable in that clause, and so on.

If the algorithm terminates with a truth assignment, it clearly returns a correct answer. Assume for now that the formula is satisfiable and that the algorithm will actually run as long as necessary to find a satisfying truth assignment.

We are mainly interested in the number of iterations of the while-loop executed by the algorithm. We refer to each time the algorithm changes a truth assignment as a step.

Since a formula has $O(n^2)$ distinct clauses, each step can be executed in $O(n^2)$ time.
• Let $S$ represent a satisfying assignment for the $n$ variables and let $A_i$ represent the variable assignment after the $i$th step of the algorithm.
• Let $X_i$ denote the number of variables in the current assignment $A_i$ that have the same value as in the satisfying assignment $S$.
• When $X_i = n$, the algorithm terminates with a satisfying assignment.
• In fact, the algorithm could terminate before $X_i$ reaches $n$ if it finds another satisfying assignment, but for our analysis the worst case is that the algorithm only stops when $X_i = n$.

Starting with $X_i < n$, we consider how $X_i$ evolves over time, and in particular how long it takes before $X_i$ reaches $n$.
• First, if $X_i = 0$ then, for any change in variable value on the next step, we have $X_{i+1} = 1$.
• Hence $\Pr(X_{i+1} = 1 \mid X_i = 0) = 1$.
• Suppose now that $1 \leq X_i \leq n - 1$.
• At each step, we choose a clause that is unsatisfied.
• Since $S$ satisfies the clause, that means that $A_i$ and $S$ disagree on the value of at least one of the variables in this clause.
Because the clause has no more than two variables, the probability that we increase the number of matches is at least $1/2$:

- the probability that we increase the number of matches could be 1 if we are in the case where $A_i$ and $S$ disagree on the value of both variables in this clause.

It follows that the probability that we decrease the number of matches is at most $1/2$.

Hence, for $1 \leq j \leq n - 1$,

$$\Pr(X_{i+1} = j + 1 | X_i = j) \geq 1/2$$

$$\Pr(X_{i+1} = j - 1 | X_i = j) \leq 1/2$$

The stochastic process $X_0, X_1, X_2, \ldots$ is not necessarily a MC:

- The probability that $X_i$ increases could depend on whether $A_i$ and $S$ disagree on one or two variables in the unsatisfied clause.

This, in turn, might depend on the clauses that have been considered in the past.

Consider the following Markov chain $Y_0, Y_1, Y_2, \ldots$:

$$Y_0 = X_0$$

$$\Pr(Y_{i+1} = 1 | Y_i = 0) = 1$$

$$\Pr(Y_{i+1} = j + 1 | Y_i = j) = 1/2$$

$$\Pr(Y_{i+1} = j - 1 | Y_i = j) = 1/2$$
• The MC $Y_0, Y_1, Y_2, \ldots$ is a pessimistic version of the stochastic process $X_0, X_1, X_2, \ldots$
  – Whereas $X_i$ increases at the next step with probability $\geq 1/2$, $Y_i$ increases with probability $= 1/2$
• It is therefore clear that the expected time to reach $n$ starting from any point is larger for the MC $Y$ than for the process $X$, and we use this fact hereafter
• This MC models a random walk on an undirected graph $G$

• The vertices of $G$ are the integers $0, \ldots, n$ and, for $1 \leq i \leq n - 1$, node $i$ is connected to node $i - 1$ and node $i + 1$
• Let $h_j$ be the expected number of steps to reach $n$ when starting from $j$
• $h_j$ is an upper bound on the expected number of steps to fully match $S$ when starting from a truth assignment that matches $S$ in $j$ locations
• Clearly, $h_n = 0$ and $h_0 = h_1 + 1$, since from $h_0$ we always move to $h_1$ in one step
• We use linearity of expectations to find an expression for other values of $h_j$
Let $Z_j$ be a RV representing the number of steps to reach $n$ from state $j$.

Now consider starting from state $j$, $1 \leq j \leq n - 1$.

With probability $1/2$, the next state is $j - 1$, and in this case $Z_j = 1 + Z_{j-1}$.

With probability $1/2$, the next step is $j + 1$, and $Z_j = 1 + Z_{j+1}$.

Hence, 

$$E[Z_j] = E\left[\frac{1}{2}(1 + Z_{j-1}) + \frac{1}{2}(1 + Z_{j+1})\right]$$

But $E[Z_j] = h_j$ and so, by applying the linearity of expectations, we obtain

$$h_j = \frac{h_{j-1} + 1}{2} + \frac{h_{j+1} + 1}{2} = \frac{h_{j-1}}{2} + \frac{h_{j+1}}{2} + 1$$

We therefore have the following system of equations:

$$h_n = 0;$$

$$h_j = \frac{h_{j-1}}{2} + \frac{h_{j+1}}{2} + 1, 1 \leq j \leq n - 1$$

$$h_0 = h_1 + 1$$

We can show inductively that, for $0 \leq j \leq n - 1$, 

$$h_j = h_{j+1} + 2j + 1$$
• It is true when $j = 0$, since $h_1 = h_0 - 1$
• For other values of $j$, we use the equation
$$h_j = \frac{h_{j-1} + h_{j+1}}{2} + 1$$
to obtain
$$h_{j+1} = 2h_j - h_{j-1} - 2$$
$$= 2h_j - (h_j + 2(j - 1) + 1) - 2$$
$$= h_j - 2j - 1$$
using the induction hypo in the second line
• We can conclude that
$$h_0 = h_1 + 1 = h_2 + 1 + 3 = \cdots = \sum_{i=0}^{n-1} 2i + 1 = \frac{n^2}{2}$$

**Lemma 7.1:** Assume that a 2-SAT formula with $n$ variables has a satisfying assignment and that the 2-SAT algorithm is allowed to run until it finds a satisfying assignment. Then the expected number of steps until the algorithm finds an assignment is at most $n^2$.

**Theorem 7.2:** The 2-SAT algorithm always returns a correct answer, if the formula is unsatisfiable. If the formula is satisfiable then with probability at least $1 - 2^{-m}$ the algorithm returns a satisfying assignment. Otherwise, it incorrectly returns that the formula is unsatisfiable.