ADVANCED ALGORITHMS (X)

CHIHAO ZHANG

1. Monte Carlo Algorithms

The *Monte Carlo* algorithms are a family of randomized algorithms based on repeated sampling. Today we first give you some examples of these algorithms and then show how Markov chains naturally arise in this setting. Let us start with the following experiment to estimate the value of π:

(1) Sample points in the square $[-1,1]\times[-1,1]$ on $\mathbb{R}^2$.
(2) Count the number of points locating in the circle of radius one centered at the origin.

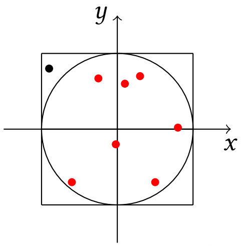

Figure 1. Monte Carlo algorithm to estimate π

Formally, for every $i = 1,2,\dots$, we independently sample a pair (X_i, Y_i) where X_i, Y_i are uniform reals on $[-1,1]$ and let

$$Z_i = \begin{cases} 1 & \text{if } X_i^2 + Y_i^2 \le 1; \\ 0 & \text{otherwise.} \end{cases}$$

Then $\mathbf{E}[Z_i] = \mathbf{Pr}[Z_i = 1] = \frac{\pi}{4}$. Therefore, if we sample m points $(X_1, X_2), \dots, (X_m, X_m)$, then the expectation of $\frac{\sum_{i=1}^m Z_i}{m}$ is $\frac{\pi}{4}$. Intuitively, the larger m we choose, more accurate the estimation is. The relation between m and the accuracy is provided by the Chernoff bound in a very tight sense.

Proposition 1. *Let $X = \sum_{i=1}^m X_i$ where each $X_i \sim \mathrm{Bin}(p)$ is an independent Bernoulli random variable. Let $\mu = pm$ be the expectation of X and $0 \le \varepsilon \le 1$ be a constant. Then*

$$\mathbf{Pr}\left[|X-\mu| \ge \varepsilon\mu\right] \le 2\exp\left(-\frac{\varepsilon^2\mu}{3}\right).$$

In our setting, we have $p = \frac{\pi}{4}$. Let $Z = \sum_{i=1}^m Z_i$. For every $0 \le \delta \le 1$, if we want

$$\mathbf{Pr}\left[\left|Z - \frac{\pi}{4}m\right| \ge \varepsilon\cdot\frac{\pi}{4}m\right] \le 2\exp\left(-\frac{\varepsilon^2\pi m}{12}\right) \le \delta,$$

then it is required that

$$m \ge \frac{12}{\varepsilon^2\pi}\log\frac{2}{\delta}.$$

In fact, the bound for the number of trials m above is proportional to $\frac{1}{p}$, which is the expected number of samples that one needs before $Z_i = 1$ for the first time.

The dependency of $\frac{1}{p}$ is crucial in many cases. Consider the problem of counting the number of solutions of DNFs. Let V be a finite set of variables. A DNF formula is of the form

$$\phi = \bigvee_{i=1}^m C_i,$$

where each $C_i = \bigwedge_{j=1}^{\ell_i} x_j$. Here x_j can be either a variable or the negation of a variable in V. We are interested in the following computational problem

#DNF	
Input:	A DNF $\phi = C_1 \vee C_2 \cdots \vee C_m$ where each $C_i = x_{i,1} \wedge x_{i,2} \cdots \wedge x_{i,\ell_i}$.
Problem:	Compute the number of satisfying assignments of ϕ.

We use $\#\phi$ to denote the number of satisfying assignments of ϕ. The problem of computing $\#\phi$ is known to be $\#\mathbf{P}$-hard, so we do not expect any polynomial-time algorithm to solve it *exactly*. We now propose a Monte Carlo algorithm to approximate the number.

The first idea would be to mimic our strategy to approximate π: Randomly sample an assignment $\sigma \in \{\text{true}, \text{false}\}^V$, and let Z_i be the indicator of whether σ satisfies ϕ. We then use $Z = \sum_{i=1}^{m}$ for sufficiently large m as an estimate. We already argued that the number of trials we need to guarantee good accuracy is proportional to $\mathbf{Pr}\left[Z_i = 1\right]^{-1}$, which in our case is exactly $\frac{2^{|V|}}{\#\phi}$. This ratio, $\frac{2^{|V|}}{\#\phi}$, can be exponential in $|V|$ when each clause C_i of ϕ is large.

In fact, we can improve the efficiency of our algorithm by sampling from another space, which can be much smaller than the set of all assignments $\{\text{true}, \text{false}\}^V$ in many cases.

For every $i \in [m]$, we use $S_i \subseteq \{\text{true}, \text{false}\}^V$ to denote the set of satisfying assignments of the clause C_i. We will sample from $\bigsqcup_{i\in[m]} S_i$, the *disjoint* union of S_i for $i \in [m]$. To ease the presentation, we let

$$\mathcal{S} = \left\{(\sigma, i) \;:\; \sigma \in S_i\right\}.$$

Obviously $\mathcal{S}$ is just another way to represent $\sqcup_{i\in[m]} S_i$. The set we really want to sample from is the set of satisfying assignments of ϕ, namely $\bigcup_{i\in[m]} S_i$. It is easy to identify a subset of $\mathcal{S}$ whose cardinality is the same as $\bigcup_{i\in[m]} S_i$. For example, we can let

$$\mathcal{T} = \left\{(\sigma, i) \in \mathcal{S} \;:\; \text{for every } j < i, \sigma \notin S_j\right\}.$$

Then $|\mathcal{T}| = \left|\bigcup_{i\in[m]} S_i\right|$ (check this!). We have the following facts:

- We can compute the size of S_i and sample from S_i u.a.r. efficiently. In fact, if S_i contains both x and $\bar{x}$ for some variable x, then it is not satisfiable. Otherwise, if it contains k literals, then $|S_i| = 2^{|V|-k}$ and sampling from S_i is trivial.
- We can sample from $\mathcal{S}$ u.a.r. efficiently. We can achieve this by first sampling an index $i \in [m]$ with probability $\frac{|S_i|}{\sum_{j\in[m]} |S_k|}$ and then uniformly sampling an assignment σ from S_i. Please verify that the pair (σ, i) we obtained in this way is uniform in $\mathcal{S}$ by yourself.
- For every pair (σ, i), we can efficiently check whether it is in $\mathcal{T}$. We simply check for every $j < i$, whether it holds that $\sigma \in S_j$.

The above facts already indicated our new Monte Carlo algorithm: Randomly sample a pair $(\sigma, i) \in \mathcal{S}$ and let Z_j be the indicator of whether (σ, i) is in $\mathcal{T}$. Since every assignment $\sigma \in \bigcup_{i\in[m]} S_i$ can appear in at most m pairs in $\mathcal{S}$, we have

$$\frac{|\mathcal{T}|}{|\mathcal{S}|} \geq \frac{1}{m}.$$

Then by the Chernoff bound we just used, it holds that if we choose $Z = \sum_{j=1}^{t} Z_j$ for $t = \Theta\left(m\right)$, the quantity $\frac{Z}{t} \cdot |\mathcal{S}|$ is a good estimate for $\#\phi$.

2. Markov Chain Monte Carlo

As shown by the examples in the last section, sampling from a given distribution is an important yet non-trivial computational task. Markov chains turn out to be a powerful tool for this kind of problems, as it is relatively easy to construct a well-behaved chain whose stationary distribution is *any* given distribution.

Let Ω be the state space and π be a distribution over Ω. The goal of this section is to define a Markov chain, or equivalently a random walk on Ω whose stationary distribution is π. We also assume an transition graph of the chain (with transition probability unknown) and an upper bound on the maximum degree of the transition graph for efficiency consideration. This is to say, for every state $x \in \Omega$, we have a set $N(x)$ called *neighbours* of x, consisting of those states that one can move from x in one step. Moreover, $\left|N(x)\right| < M$ for every $x \in \Omega$. In fact, this assumption is quite natural. In most applications, the space Ω is very large (say, exponential in the size of the input), and the philosophy behind the MCMC is that we can approximate π by only sampling from a small space (a local distribution on $N(x)$) in each step. Otherwise, the construction of the Markov chain is trivial as we can always let $P = \Pi = \begin{bmatrix} \pi^T \\ \vdots \\ \pi^T \end{bmatrix}$, and simulating the chain is almost equivalent to directly sampling from π (which is not affordable)!

We now propose a Markov chain called *Metropolis Algorithm* to sample from π. Assuming now we are at some state $x \in \Omega$, the algorithm determines the next state in two steps:

1. First sample a state y from $N(x) \cup \{x\}$ so that every $z \in N(x)$ is of probability $\frac{1}{M}$ and the state x is of probability $1 - \frac{|N(x)|}{M}$.
2. Then move to state y with probability $\min\left\{1, \frac{\pi(y)}{\pi(x)}\right\}$ (if fail to move, just stay at x).

Proposition 2. *If the pre-specified transition graph is connected (irreducible) and symmetric ($x \in N(y)$ iff $y \in N(x)$ for every $x, y \in \Omega$), then Metropolis algorithm is an aperiodic reversible chain with stationary distribution π.*

Proof. Let P be the transition matrix. Then it is clear that $P(x,x) > 0$ for every x since $M > \left|N(x)\right|$. So P is aperiodic. Then it suffices to verify the detailed balance condition. For every x, y such that $y \in N(x)$, assuming $\pi(y) \le \pi(x)$, we have

$$\pi(x)P(x,y) = \pi(x) \cdot \frac{1}{M} \cdot \min\left\{1, \frac{\pi(y)}{\pi(x)}\right\} = \frac{\pi(y)}{M} = \pi(y) \cdot \frac{1}{M} \cdot \min\left\{1, \frac{\pi(x)}{\pi(y)}\right\} = \pi(y)P(y,x).$$

□

Now let me show you some applications of Metropolis algorithms. Consider the following *hardcore model* in statistical physics.

- We are given an undirected graph $G(V,E)$ with a parameter λ.
- Consider the set of independent sets $\mathcal{I}$ of G. Each independent set $I \in \mathcal{I}$ is associated with a weight $w(I) = \lambda^{|I|}$.
- The *Gibbs measure* μ of the model is a distribution over $\mathcal{I}$ such that for every $I \in \mathcal{I}$,

$$\mu(I) = \frac{w(I)}{Z},$$

  where $Z = \sum_{I \in \mathcal{I}} w(I)$ is the normalizing factor.

We can use the Metropolis algorithm to sample from μ. The state space Ω is $\mathcal{I}$, the set of all independent set. First we setup the transition graph. For each $I \in \mathcal{I}$, we are allowed to move to any independent set that differs from I by exactly one vertex. Formally, we define $N(I)$ as a set of independent sets such that $I' \in N(I)$ if and only if $I' = I \cup \{u\}$ for some $u \in V \setminus I$ or $I' = I \setminus \{u\}$ for some $u \in I$. Therefore, we have $\left|N(I)\right| \le |V|$ and can let $M = |V| + 1$. Assuming now we are at some independent set I, we then faithfully follow the steps of the Metropolis algorithm to find the next independent set:

1. With probability $\frac{1}{M}$, stay at I and terminate.
2. Choose a vertex $u \in V$ uniformly at random.
3. If $u \in I$, move to $I \setminus \{u\}$ with probability $\min\left\{1, \frac{1}{\lambda}\right\}$. If $u \notin I$ and $I \cup \{u\} \in \mathcal{I}$ is an independent set, move to $I \cup \{u\}$ with probability $\min\{1, \lambda\}$.
4. If the above step fails to move, stay at I.

Proposition 2 implies that the above chain converges to the Gibbs measure.