Randomness and Computation

or, “Randomized Algorithms”

Mary Cryan

School of Informatics
University of Edinburgh

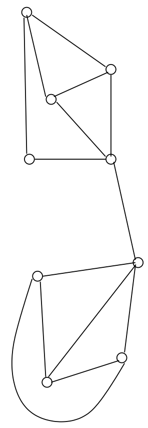
Hamiltonian cycles

Definition

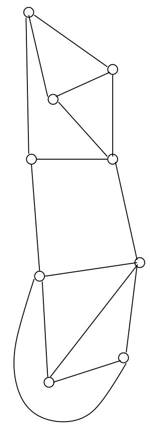
Given an undirected graph $G = (V, E)$ with $V = [n]$, an *Hamiltonian circuit (HC)* of G is a permutation π of the vertex set $[n]$ such that for every $i \in [n]$, we have $(v_{\pi(i)}, v_{\pi(i+1)}) \in E$ (with $n + 1$ identified with 1).

- A particular graph G may have many HCs, or in some cases (more likely for a sparse graph) no HC.
- Similar definition holds for a *directed graph* (we require $(v_{\pi(i)} \to v_{\pi(i+1)})$ for every $i \in [n]$).
- The problem of deciding whether a given graph contains a HC is *NP-complete* (also *NP-complete* for directed graphs). So we *don't* expect there is a deterministic polynomial-time algorithm to *find* a HC in an arbitrary graph (or to *decide* whether there is one).

Examples of Hamiltonian Circuits (or not)



No HC



(more than one) HC

Intuitively HCs are more likely in denser graphs

Random Graphs - Erdős-Rényi model $\mathcal{G}_{n,p}$

n vertices.

Some fixed probability p.

For every $i \in [n]$, every $j \in [n] \setminus \{i\}$

- We flip a coin with biased probability p, add (i, j) to E if the flip is successful, don't add it otherwise.

All the (i, j) trials are *identical* (probability p) and *independently* distributed.

$\mathbf{E}_{n,p}[|E|] = \sum_{i,j \in [n], i \neq j} \Pr[(i, j) \in E] \;=\; \frac{n(n-1)}{2} p.$

Expected degree of any vertex i is $(n - 1)p$.

Can use *deferred decisions* to analyse algorithms/structures on $\mathcal{G}_{n,p}$.

Hamilton cycles in Erdős-Rényi graphs

Theorem (Komlós and Szemerédi (1983))

Suppose we generate G according to $\mathcal{G}_{n,p}$. Then the existence of a HC in G is characterised by the value of p in relation to n:

$$\Pr_{n,p}[G \text{ has a HC}] \rightarrow \begin{cases} 0 & \text{if } pn - \ln(n) - \ln\ln(n) \rightarrow -\infty \\ 1 & \text{if } pn - \ln(n) - \ln\ln(n) \rightarrow +\infty \\ e^{-e^{-c}} & \text{if } pn - \ln(n) - \ln\ln(n) \rightarrow c \end{cases}$$

(in the final case, c being any constant)

- These are all *with high probability* results, holding with probability $1 - o(1)$ (tending to 1 as $n \rightarrow \infty$).
- This kind of result is described as a *sharp threshold*.
- There is a polynomial-time algorithm to *find the HC* in the middle case (Bollobás, Fenner and Frieze, 1985).
- We **will not** prove Komlós &Szemerédi's result, we will show how to *find a HC* for $p \geq \frac{40\ln(n)}{n}$ (easier).

Hamilton cycles in Erdős-Rényi graphs - Extend or Rotate

Given a graph $G = (V, E)$ generated according to $\mathcal{G}_{n,p}$.

Algorithm maintains a current path P of the form $v_1, \ldots, v_k$, the current head (*hd*) is v_k.

Intuitively, the algorithm tries to randomly choose an *extension edge* (adjacent to the head) which adds a new vertex to the path (if not, we then "Rotate").

Three operations are used to "grow" a path (from a starting vertex):

- "Reverse"
- "Rotate"
- "Extend"

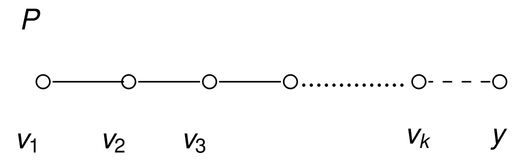
UnUsed(v) contains the Adjacent edges to v which have not been used to extend *from* v (originally all adjacent edges).

Assume *UnUsed*(v) is randomly shuffled for each v.

Hamilton cycles in Erdős-Rényi graphs - Extend

Algorithm maintains a current path P of the form $v_1, \ldots, v_k$, the current head (*hd*) is v_k.

The algorithm chooses the "next" *extension edge* from *UnUsed*(v_k) (adjacent to the head), hoping this will add a new vertex to the path.

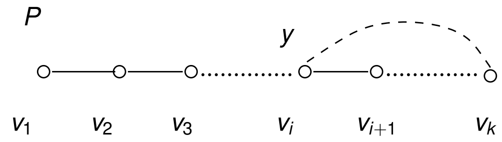


(v_k, y) in G, and y not in P => "extend"

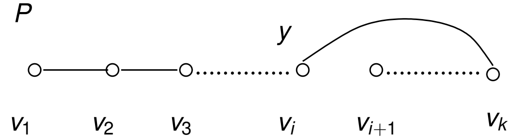
Ideal case. Note that (v_k, y) is a uniform random choice from *Adj*(v_k), as the edges were shuffled at the start.

Hamilton cycles in Erdős-Rényi graphs - Rotate

Sometimes the randomly chosen extension "loops back" onto P (less ideal). We don't iterate through other v_k neighbours, we "Rotate".



(v_k, y) in G, but y already in P => "rotate"



Add (v_k, v_i), delete (v_i, v_{i+1}), v_{i+1} is the new "hd"

“Reverse, Extend or Rotate” (Algorithm 5.2)

Algorithm ReverseExtendRotate$(G = (V, E))$

1. **for** $v \in V$ **do**
2. $\quad$ $Used(v) \leftarrow \{\}$, $UnUsed(v) \leftarrow \{(v, u) : u \in Adj_G(v)\}$.
3. Initialise P with a uniform random vertex, initialise hd also.
   // *Throughout P is some $v_1 \dots v_k$ (distinct vertices), hd is v_k* //
4. **while** (P is not a HC and $UnUsed(hd) \neq \{\}$) **do**
5. $\quad$ With prob. $\frac{1}{n}$, “Reverse” P (and reset $hd \leftarrow v_1$)
6. $\quad$ With prob. $\frac{|Used(v_k)|}{n}$, choose $(v_k, v_i) \in_{uar} Used(hd)$, and “Rotate” (and reset $hd \leftarrow v_{i+1}$).
7. $\quad$ With prob. $1 - \frac{(1+|Used(v_k)|)}{n}$, take the *first* edge (v_k, y) from $UnUsed(hd)$, and “Extend” or “Rotate” (depends on y).
   Move (v_k, y) from $UnUsed(v_k)$ to $Used(v_k)$.
   Update hd to either y (Extend) or v_{i+1} (Rotate).
8. Check whether P is a HC and return P or “no".

Analysing Algorithm 5.2

- Overall, the key operation for building the HC is “Extend”. “Rotate” and “Reverse” are helper operations which help the *analysis* go through (well, “Rotate” also helps us get unstuck).
- Asking quite a lot to get a full HC on a “run” where we more-or-less just add random edges to extend P. So we’ll need to run the loop for a super-linear number of steps ($\Omega(n \ln(n))$, see Theorem 5.16, Corollary 5.17).
- Assume for Lemma 5.15, Thm 5.16 that $UnUsed(v)$ is randomly generated by adding every possible (u, v) with probability q, in random order (can use “*deferred decisions*” in analysis).
  - All the $UnUsed(\cdot)$ sets are *assumed to be independent* for proving Lemma 5.15 and Theorem 5.16 (not strictly true, fixed in Cor 5.17). Means to an end ...

Lemma 5.15

Lemma (Lemma 5.15)

Supposed we run Algorithm ReverseExtendRotate *on $G = (V, E)$ with the $UnUsed(v)$ sets generated independently with probability q for each possible neighbour, and random orders. Let V_t be the “hd” vertex after t steps. Then, as long as $UnUsed(V_t) \neq \{\}$, for any $u \in V$,*

$$\Pr[V_{t+1} = u \mid V_t = u_t, \dots, V_0 = u_0] \; = \; \frac{1}{n}.$$

Proof.

Identical to book.
Easy for v_1, and for any of the P-vertices v_{i+1} such that $v_i \in Used(hd)$.
For other u vertices (on P or otherwise), we use the principle of deferred decisions, plus the assumptions on slide 10. $\square$

Theorem 5.16

Theorem (Theorem 5.16)

Supposed we run Algorithm ReverseExtendRotate *on G with the $UnUsed(v)$ sets generated independently with probability $q \geq \frac{20\ln(n)}{n}$ for each possible neighbour, and random orderings. Then the algorithm finds a HC after $O(n\ln(n))$ iterations of the loop at 4., with probability $1 - O(n^{-1})$.*

Proof Failure after $3n\ln(n)$ iterations means that *either*

- $\mathcal{E}_1$: We did $3n\ln(n)$ iterations without constructing a HC, with all $UnUsed(hd)$ sets staying non-empty, *or*;
- $\mathcal{E}_2$: At least one of the $UnUsed(hd)$ lists became empty during the $3n\ln(n)$ iterations.

To bound $\Pr[\mathcal{E}_1]$, we want the prob. of *not* finding a HC, when at each step, the next “*hd*” is uniform from V (guaranteed by Lemma 5.15).

This is the “coupon collector" problem (must hit all n vertices).

Theorem 5.16 cont'd.

Proof cont'd. For event $\mathcal{E}_1$, probability that any *particular* v does not become "*hd*" at some stage over $2n\ln(n)$ iterations, is at most

$$\left(1-\frac{1}{n}\right)^{2n\ln(n)} < e^{-2\ln(n)} = \frac{1}{n^2}.$$

Probability that *some* v fails to become *hd* in this window, is at most $\frac{1}{n}$ by the Union Bound.

Need to complete the HC with a closing edge to v_1, over remaining $\ln(n)n$ steps. Probability of failure is at most $(1-\frac{1}{n})^{n\ln(n)} < \frac{1}{n}$.

So $\Pr[\mathcal{E}_1] \le \frac{2}{n}$.

For event $\mathcal{E}_2$, we partition into:

$\mathcal{E}_{2a}$: Some v had at least $9\ln(n)$ edges removed from $UnUsed(v)$ during the $3n\ln(n)$ steps. *or*,

$\mathcal{E}_{2b}$: Some v originally had $\le 10\ln(n)$ edges in $UnUsed(v)$.

Theorem 5.16 cont'd.

Proof cont'd.
$\mathcal{E}_{2b}$ first. For a specific v, expected degree is $20\frac{\ln(n)}{n}(n-1)$, so $\ge 19\ln(n)$ for reasonable n. By Chernoff's bounds (Thm 4.5, 2.)

$$\Pr[|UnUsed(v)| \le 10\ln(n)] \le e^{-19\cdot 9^2\ln(n)/(2\cdot 19^2)} = e^{-2.25\ln(n)} \le \frac{1}{n^2}.$$

Hence by the Union Bound (all n vertices) $\Pr[\mathcal{E}_{2b}] \le \frac{1}{n}$.

$\mathcal{E}_{2a}$ next. For a specific v, edge removal from $UnUsed(v)$ can only happen when $hd = v$, probability $\frac{1}{n}$ each step. Let the number of *hd* roles for v be the random variable HD_v. HD_v is distributed as $B(3n\ln(n), \frac{1}{n})$, with $\mathbf{E}[HD_v] = 3\ln(n)$.
By Chernoff's bounds (Thm 4.4, 2.),

$$\Pr[|HD_v| \ge 9\ln(n)] \le e^{-3\ln(n)2^2/3} = e^{-4\ln(n)} = \frac{1}{n^4}.$$

Hence by the Union Bound (all n vertices) $\Pr[\mathcal{E}_{2a}] \le \frac{1}{n}$.

Hence $\Pr[\mathcal{E}_1 \cup \mathcal{E}_2] \le \frac{4}{n}$, as required. □

Corollary 5.17

Proof of Theorem 5.16 assumed that the $UnUsed(v)$ sets were all generated independently of each other when they are randomly populated. In the "real world" of $\mathcal{G}_{n,p}$, of course (u, v) would add an entry into *two* "*UnUsed*" sets.

Our analysis (essentially) assumes that *either* $(u, v) \in UnUsed(u)$ or $(u, v) \in UnUsed(v)$ is ok to have the edge in the HC.

See Corollary 5.17 for how to define p so that we can populate the *UnUsed* lists randomly and independently, with $q \ge 20\ln(n)/n$.

Reading and Doing

Next topics coming up is the Probabilistic method and Derandomization, could read some of Chapter 6 to prepare.

Exercises:

- Exercises 5.3 and 5.7 are good reminders of the basic "balls in bins" ideas.
- Analysis at top of slide 13 is essentially the "coupon collector" problem, done for specific v, then Union Bound. Compare this to what we got with $2n\ln(n)$ on slide 16 of lecture 6. Surprised?
- Read Corollary 5.17 and understand the details.
- Exploratory exercise on "marking the binary tree" (section 5.8).