1. Connectivness of Graphs

Def. 1. Vertices $u, v \in V, u \neq v$ are called **connected** if there is a $u-v$-path in G.

Def. 2. A graph $G = (V, E)$ is **connected** if $\forall u, v \in V, u \neq v$ a $u-v$-path in G exists.

Def. 3. Vertices $u, v \in V, u \neq v$ are called k-**connected** if $|V| \geq k+1$ and

$$\forall X \subseteq V \setminus \{u, v\}$$

$$\text{with } |X| < k,$$

$$u \text{ and } v \text{ are connected in } G[V \setminus X]$$

Def. 4. A graph $G = (V, E)$ is called k-**connected** if $|V| > k$ and

$$\forall X \subseteq V \text{ with } |X| < k$$

$$G[V \setminus X] \text{ is connected}$$

Def. 5. $X \subseteq V \setminus \{u, v\}$ is called a $u-v$-**separator** if u and v are in different connected components of $G[V \setminus X]$.

Rem. 6. If G contains a vertex v with $deg(v) < k$, then G cannot be k-connected.

Thm. 7. Menger's Theorem (Vertex Version): The minimum size of a $u-v$-separator is equal to the maximum number of internally vertex-disjoint $u-v$-paths. A graph is k-connected iff $\forall u \neq v$ there are k internally vertex-disjoint $u-v$-paths.

Def. 8. A graph $G = (V, E)$ is called k-**edge-connected** if

$$\forall X \subseteq E \text{ with } |X| < k$$

$$(V, E \setminus X) \text{ is connected}$$

Thm. 9. Menger's Theorem (Edge Version): A graph G is k-edge-connected iff. $\forall u, v \in V, u \neq v$ there are k edge-disjoint $u-v$-paths.

Thm. 10. It always holds that:

$$\text{Vertex Connectivity} \leq \text{Edge Connectivity}$$

$$\leq \text{Minimum Degree } \delta(G)$$

Lets look at the special case where $k = 1$:

Def. 11. For a connected graph $G = (V, E)$:

- Vertex $v \in V$ is a **cut vertex** iff. $G[V \setminus \{v\}]$ is disconnected.
- Edge $e \in E$ is a **bridge** iff. $(V, E \setminus \{e\})$ is disconnected.

Lem. 12. If $e = \{x, y\} \in E$ is a bridge, then $\deg(x) = 1$ or x is a cut vertex.

Def. 13. Blocks: Define an equivalence relation $\sim$ on E:

$$e \sim f \iff e = f \lor \exists \text{ cycle containing } e \text{ and } f$$

The equivalence classes are called **blocks**.

Note: Two blocks can share at most one vertex, which must be a cut vertex.

Def. 14. Block Graph: Bipartite graph $T = (A \cup B, E_T)$ where: $A = \{\text{cut vertices}\}$ $B = \{\text{blocks}\}$ Edge $\{a, b\} \in E_T \iff a$ is incident to an edge in block b.

Thm. 15. If G is connected, the block graph of G is a tree.

Algorithm 16

```
#DFS & Low Values for cut vertices/bridges in O(|V|+|E|)
low[v] := #smallest DFS number reachable from v by tree edges and <=1 back edge.

low[v] = min{ dfs[v], min_{(v,w) back} dfs[w], min_{(v,w) tree} low[w] }

#A vertex v is a cut vertex iff:
v != root and exists child u with low[u] >= dfs[v].
v = root and has >= 2 children in DFS tree.

#A tree edge (v,w) is a bridge iff:
low[w] >= dfs[v].
```

2. Hamiltonian Cycles and Eulertours

2.1 Eulertours

Def. 17. An **Eulertour** is a closed walk in a graph G that contains every edge exactly once.

Thm. 18. A connected graph $G = (V, E)$ contains an Eulertour iff the degree of every vertex is even.

Algorithm 19

```
#Find Eulertour in O(|E|) (Runner and Turtle)
Runner traverses until hitting a visited vertex (cycle found).
Turtle follows path until an unvisited neighbor is found.
Runner is set loose again from there, expanding the tour by cycles.
```

2.2 Hamiltonian Cycles

Def. 20. A **Hamiltonian Cycle** is a cycle containing every vertex exactly once.

Thm. 21. For $m, n \geq 2$, an $n \times m$ grid has a Hamiltonian cycle iff $n \cdot m$ is even.

Thm. 22. A bipartite graph $G = (A \cup B, E)$ with $|A| \neq |B|$ cannot contain a Hamiltonian cycle.

Thm. 23. Dirac's Theorem: Every graph G with $|V| \geq 3$ and minimum degree $\delta(G) \geq |V|/2$ contains a Hamiltonian cycle.

Algorithm 24

```
#DP for Hamiltonian Cycle O(n^2 * 2^n)
#P[S, x] = 1 if exists 1-x path containing exactly vertices in S.
P[S, x] = max { P[S \setminus {x}, y] | y in N(x) cap S, y != 1 }
```

2.2.1 Counting Hamiltonian Cycles

Def. 25. For $S \subseteq V \setminus \{s\}$, define:

$$W_S := \{\text{Walks of length } n \text{ from } s \text{ to } s \text{ avoiding } S\}$$

$|W_S|$ is the (s, s)-entry of $(A_S)^n$, where A_S is the adjacency matrix of $G[V \setminus S]$.

Thm. 26. The number of Hamiltonian cycles in G is:

$$\frac{1}{2} \sum_{S \subseteq V \setminus \{s\}} (-1)^{|S|} |W_S|$$

This can be computed in $O(n^{2.81} \log n \cdot 2^n)$ time and $O(n^2)$ space (vs. $O(n \cdot 2^n)$ space for DP).

Thm. 27. Hamiltonian decision is NP-Complete (Karp 1972).

2.3 Travelling Salesman Problem (TSP)

The TSP finds a Hamiltonian cycle in K_n with minimal weight (shortest roundtrip).

Thm. 28. The TSP is NP-Complete (reducible from Hamiltonian Cycle).

2.3.1 Metric TSP

For graphs where triangle inequality holds:

$$w(u, v) \leq w(u, w) + w(w, v)$$

Algorithm 29

```
#2-Approximation for Metric TSP
1. Determine MST T (|E(T)| <= |E(H)| for any Hamiltonian cycle H)
2. Double edges of T to get Eulerian multigraph T'
3. Traverse Eulertour, skip already visited vertices
#Result: length <= 2 * |E(T)|
```

Algorithm 30

```
#Christofides 1.5-Approximation for Metric TSP
1. Determine MST T (|E(T)| <= |E(H)|)
2. Find minimal matching M for odd-degree vertices of T (|M| <= |E(H)|/2)
3. Determine Eulertour in T + M
#Result: |W| <= |E(T)| + |M| <= 1.5 * |E(H)|
```

3. Matchings

Def. 31. An edge set $M \subseteq E$ is a **matching** in $G = (V, E)$ if no vertex is incident to more than one edge from M.

- **Covered**: Vertex v is incident to an edge in M.
- **Perfect**: Every vertex is covered ($|M| = |V|/2$).
- **Maximal**: Cannot be extended by adding edges.
- **Maximum**: No matching with more edges exists.

Algorithm 32

```
#Greedy Algorithm (Leads to Maximal Matching)
#Size of M is at least half the size of a maximum matching.
M = empty set
While E is not empty:
Choose arbitrary edge e from E, add to M
Remove all edges incident to e's endpoints from E
```

Thm. 33. Hall's Marriage Theorem: A bipartite graph $G = (A \cup B, E)$ contains a matching of cardinality $|M| = |A|$ iff:

$$\forall X \subseteq A : |X| \leq |N(X)|$$

Thm. 34. Frobenius Corollary: Every k-regular bipartite graph contains a perfect matching. In fact, it is a union of perfect matchings. A perfect matching can be found in $O(|E|)$ time if the graph is k- or 2^k-regular.

Def. 35. An M**-augmenting path** alternates between edges in M and not in M, starting and ending in uncovered vertices.

Thm. 36. Berge's Theorem (1957): A matching M is maximum iff there is no M-augmenting path.

Algorithm 37

```
\textbf{Hopcroft-Karp} ($O(|V|^{1/2}|E|)$
        time for bipartite graphs): While there
         is an augmenting path, find a maximal
        set of vertex-disjoint shortest
        augmenting paths using BFS, and augment
         $M$ along all of them.
```

4. Coloring

Def. 38. A k**-coloring** of $G = (V, E)$ is a mapping $c : V \to [k]$ such that $c(u) \neq c(v)$ for all $\{u, v\} \in E$. A **stable set** is a set of vertices where no two are adjacent. The **chromatic number** $\chi(G)$ is the minimum number of colors. $\chi(G) \leq 2 \iff G$ is bipartite.

Thm. 39. For $k \geq 3$, the problem "Is $\chi(G) \leq k$?" is NP-complete. For general graphs all ϵ-approximations are NP-hard.

Algorithm 40

```
\textbf{Greedy Coloring} ($\le \Delta(G) + 1$
        colors):
\begin{itemize}
    \item There exists an optimal vertex
          order yielding exactly $\chi(G)$
          colors.
    \item \textbf{Heuristic}: Delete vertices
           in order of minimum degree. If
          every subgraph has a vertex of
          degree $\le k$, this yields $\le k
          +1$ colors (e.g. trees are $2$-
          colorable).
\end{itemize}
\textbf{Block-graphs}: If every block can be
        $k$-colored, the graph can be $k$-
        colored.
```

Thm. 41. Brooks' Theorem: For a connected graph G that is neither a complete graph K_n nor an odd cycle C_{2n+1}:

$$\chi(G) \leq \Delta(G)$$

Can be found in $O(|E|)$ time by: 1) If $\Delta(G) = 2$, 2-color. 2) If $\exists v, \deg(v) < \Delta(G)$ use greedy heuristic. 3) If $\exists$ cut vertex, color blocks. 4) Else remove non-adj. $x, y \in N(v)$ and color recursively.

5. Probability Theory

Def. 42. A **discrete probability space** is a set of elementary events $\Omega = \{\omega_1, \ldots, \omega_n\}$. Every event ω_i has a probability $P[\omega_i] \in [0, 1]$ such that

$$\sum_{\omega_i \in \Omega} P[\omega_i] = 1$$

Rem. 43. A finite space where every event has equal probability is a **Laplace-Space**. For any event E:

$$P[E] = \frac{|E|}{|\Omega|}$$

Def. 44. An **Event** $E \subseteq \Omega$ has probability $P[E] = \sum_{\omega_i \in E} P[\omega_i]$. The complementary event $\bar{E} = \Omega \setminus E$ has probability $P[\bar{E}] = 1 - P[E]$.

Thm. 45. For two events $A, B \subseteq \Omega$: $P[\emptyset] = 0$, $P[\Omega] = 1$, $P[A] \in [0, 1]$. If $A \subset B$ then $P[A] \leq P[B]$.

Thm. 46. Union Bound: For events $A_1, \ldots, A_n$:

$$P[\bigcup_{i=1}^{n} A_i] \leq \sum_{i=1}^{n} P[A_i]$$

(Equality iff A_i are pairwise disjoint).

Thm. 47. Inclusion-Exclusion Principle: For $n \geq 2$:

$$P[\bigcup_{i=1}^{n} A_i] = \sum_{k=1}^{n} (-1)^{k+1} \sum_{I \subset [n], |I| = k} P[\bigcap_{j \in I} A_j]$$

5.1 Conditional Probability

Def. 48. For $P[B] > 0$, the **conditional probability** of A given B is:

$$P[A \mid B] = \frac{P[A \cap B]}{P[B]}$$

Rem. 49. Always holds: $P[A \mid \Omega] = P[A]$, $P[A \mid A] = 1$, $P[A \mid \bar{A}] = 0$, $P[A \cap B] = P[A \mid B]P[B]$.

Thm. 50. Multiplication-Theorem: If $P[\bigcap_{i=1}^{n-1} A_i] > 0$:

$$P[\bigcap_{i=1}^{n} A_i] = P[A_1] \cdot P[A_2 \mid A_1]$$

$$\cdots \cdot P[A_n \mid \bigcap_{i=1}^{n-1} A_i]$$

Thm. 51. Theorem of total probability: For pairwise disjoint A_i and $B \subseteq \bigcup_{i=1}^{n} A_i$:

$$P[B] = \sum_{i=1}^{n} P[B \mid A_i] P[A_i]$$

Thm. 52. Bayes' Theorem: For exactly defined disjoint A_i, $B \subseteq \bigcup A_i$, and $P[B] > 0$:

$$P[A_i \mid B] = \frac{P[B \mid A_i] P[A_i]}{\sum_{j=1}^{n} P[B \mid A_j] P[A_j]}$$

5.2 Independence

Def. 53. Two events A, B are **independent** if:

$$P[A \cap B] = P[A]P[B] \text{ or } P[A \mid B] = P[A]$$

Events $A_1, \ldots, A_n$ are independent if for all $I \subseteq \{1, \ldots, n\}$:

$$P[\bigcap_{i \in I} A_i] = \prod_{i \in I} P[A_i]$$

Lem. 54. Let $A_1, \ldots, A_n$ be events, $A_i^0 = \bar{A}_i$, $A_i^1 = A_i$. They are independent iff $\forall (s_1, \ldots, s_n) \in \{0, 1\}^n$:

$$P[\bigcap_{i=1}^{n} A_i^{s_i}] = \prod_{i=1}^{n} P[A_i^{s_i}]$$

Lem. 55. If A, B, C are independent, then so are $A \cap B$ and C, and $A \cup B$ and C.

5.3 Random Variables

Def. 56. A **random variable** is a map $X : \Omega \to \mathbb{R}$. The **probability mass function (PMF)** is $f_X(x) = P(X = x)$. The **distribution function (CDF)** is $F_X(x) = P(X \leq x)$.

5.3.1 Expectation

Def. 57. The expectation of X is defined as:

$$\mathbb{E}[X] \coloneqq \sum_{\alpha \in W_x} \alpha P(X = \alpha)$$

Thm. 58. For a random variable X with values in $\mathbb{N}_0$:

$$\mathbb{E}[X] = \sum_{k=1}^{\infty} P(X \geq k)$$

Def. 59. Indicator-variable X_E for $E \subseteq \Omega$: $X_E(\omega) = 1$ if $\omega \in E$, else 0.

$$\mathbb{E}[X_E] = P(E)$$

Thm. 60. Linearity of Expectation: For random variables $X_1, \ldots, X_n$ and $a_1, \ldots, a_n, b \in \mathbb{R}$:

$$\mathbb{E}[\sum_{i=1}^{n} a_i X_i + b] = \sum_{i=1}^{n} a_i \mathbb{E}[X_i] + b$$

Algorithm 61

```
#Maximal Stable Set Algorithm
S[v] = 1/0 with probability p
For all {u,v} in E:
if S[u] == 1 and S[v] == 1: S[v] = 0
Expected size: $\mathbb{E}[\mid S \mid] \geq
        n*p - m*p^2$
```

5.3.2 Variance

Def. 62. For X with $\mathbb{E}[X] = \mu$, the **variance** is:

$$Var[X] \coloneqq \mathbb{E}[(X - \mu)^2]$$

Standard deviation $\sigma \coloneqq \sqrt{Var[X]}$.

Thm. 63.

$$Var[X] = \mathbb{E}[X^2] - (\mathbb{E}[X])^2$$

Lem. 64. For $a, b \in \mathbb{R}$: $Var[aX + b] = a^2 Var[X]$

Def. 65. $\mathbb{E}[X^k]$ is the k-th **moment**. $\mathbb{E}[(X - \mathbb{E}[X])^k]$ is the k-th **central moment**.

5.3.3 Multiple & Combined Variables

Def. 66. RVs $X_1, \dots, X_n$ are **independent** iff $\forall a_i \in W_{X_i}$:

$$P[X_1 = a_1, \dots, X_n = a_n] = \prod_{i=1}^{n} P[X_i = a_i]$$

Thm. 67. If X, Y are independent, $f_{X+Y}(\alpha) = \sum_\beta f_X(\beta) f_Y(\alpha - \beta)$.

Rem. 68. Calculation Rules:

- $\mathbb{E}[X+Y] = \mathbb{E}[X] + \mathbb{E}[Y]$
- If X, Y independent: $\mathbb{E}[XY] = \mathbb{E}[X]\mathbb{E}[Y]$
- If X, Y independent: $Var[X+Y] = Var[X] + Var[Y]$

Note $Var[XY] \neq Var[X]Var[Y]$ in general.

Thm. 69. Total Expectation: For pairwise disjoint events $A_1, \dots, A_n$ covering Ω:

$$\mathbb{E}[X] = \sum_{i=1}^{n} \mathbb{E}[X \mid A_i] P[A_i]$$

Thm. 70. Wald's Equation: If N and X are independent RVs and $Z \coloneqq \sum_{i=1}^{N} X_i$ (where X_i are independent copies of X):

$$\mathbb{E}[Z] = \mathbb{E}[N]\mathbb{E}[X]$$

5.4 Distributions

Def. 71. Bernoulli(p): $X \in \{0, 1\}$

$$P[X=1] = p \quad P[X=0] = 1-p$$

$$\mathbb{E}[X] = p \quad Var[X] = p(1-p)$$

Def. 72. Binomial(n, p): $X \in \{0, \dots, n\}$

$$P[X=k] = \binom{n}{k} p^k (1-p)^{n-k}$$

$$\mathbb{E}[X] = np \quad Var[X] = np(1-p)$$

Def. 73. Poisson(λ):

$$P[X=k] = \frac{e^{-\lambda}\lambda^k}{k!}$$

$$\mathbb{E}[X] = \lambda \quad Var[X] = \lambda$$

Def. 74. Geometric(p):

$$P[X=k] = p(1-p)^{k-1}$$

$$\mathbb{E}[X] = \frac{1}{p} \quad Var[X] = \frac{1-p}{p^2}$$

(Number of throws until first success).

Def. 75. Negative Binomial of order n:

$$P[X=k] = \binom{k-1}{n-1} p^n (1-p)^{k-n}$$

(Number of throws until n-th success).

5.5 Bounding Probability

Thm. 76. Markov's Inequality: For non-negative RV X and $t \geq 0$:

$$P[X \geq t] \leq \frac{\mathbb{E}[X]}{t} \quad \text{or} \quad P[X \geq t\mathbb{E}[X]] \leq \frac{1}{t}$$

Thm. 77. Chebyshev's Inequality: For any RV X and $t \geq 0$:

$$P[|X - \mathbb{E}[X]| \geq t] \leq \frac{Var[X]}{t^2}$$

5.5.1 Chernoff Bounds

Applies to sums of independent Bernoulli RVs $X = \sum X_i$ with $P[X_i = 1] = p_i$.

Thm. 78. For $0 \leq \delta \leq 1$:

$$P[X \geq (1+\delta)\mathbb{E}[X]] \leq e^{-\frac{1}{3}\delta^2 \mathbb{E}[X]}$$

$$P[X \leq (1-\delta)\mathbb{E}[X]] \leq e^{-\frac{1}{2}\delta^2 \mathbb{E}[X]}$$

For larger deviations ($t \geq 2e\mathbb{E}[X]$):

$$P[X \geq t] \leq 2^{-t}$$

5.5.2 Target Shooting (Sampling)

To estimate $|S|/|U|$, sample n elements uniformly. Let $Y = \frac{1}{n}\sum Y_i$ be the fraction of samples in S.

Thm. 79. With given $\delta, \epsilon \geq 0$, if $n \geq \frac{3|U|}{|S|\epsilon^2} \ln \frac{2}{\delta}$, then

$$Y \in \left[(1-\epsilon)\frac{|S|}{|U|}, (1+\epsilon)\frac{|S|}{|U|}\right]$$

with probability $\geq 1 - \delta$.

6. Randomized Algorithms

Def. 80. A **Randomized Algorithm** is a function $A(I, R)$ with input I and a source of randomness R.

- **Monte Carlo**: Always fast, but result not always correct.
- **Las Vegas**: Always correct, but runtime may vary.

6.1 Error Reduction

Thm. 81. If A never returns a false result but may return ??? with $P[\text{correct}] \geq \epsilon$, then calling A up to $N = \epsilon^{-1}\ln(\delta^{-1})$ times gives $P[\text{correct}] \geq 1 - \delta$.

Imp. 82. Generally **not possible to reduce error for Monte-Carlo**, except for algorithms with **one-sided error** or **error probability** $< \frac{1}{2}$.

Thm. 83. One-sided error: If $P[A(I) = \text{true}] = 1$ for true-instances and $P[A(I) = \text{false}] \geq \epsilon$ for false-instances, then $N = \epsilon^{-1}\ln(\delta^{-1})$ calls yield $P[\text{correct}] \geq 1 - \delta$.

Thm. 84. Majority vote: If $P[\text{correct}] \geq \frac{1}{2} + \epsilon$, then $N = 4\epsilon^{-2}\ln(\delta^{-1})$ calls with majority answer give $P[\text{correct}] \geq 1 - \delta$.

Thm. 85. Optimization: If $P[A(I) \geq f(I)] \geq \epsilon$, then $N = \epsilon^{-2}\ln(\delta^{-1})$ calls returning the best solution gives $P[A_\delta(I) \geq f(I)] \geq 1 - \delta$.

6.2 Primality Testing

Def. 86. Euler's Totient $\varphi(n) = |Z_n^*|$: number of integers in $\{1, \dots, n-1\}$ coprime to n. If n is prime, $\varphi(n) = n - 1$.

Thm. 87. Lagrange: If H is a subgroup of G, then $|H|$ divides $|G|$. **Fermat's Little Thm**: If n is prime, then $a^{n-1} \equiv 1 \pmod{n}$ for all $a \in \{1, \dots, n-1\}$.

Algorithm 88

```
#Fermat Test
Choose a in {1...n-1} uniformly at random.
If a^{n-1} mod n = 1: return "Prime"
else: return "Not Prime"
#If n is prime: always correct.
#If n composite (not Carmichael): error <=
      1/2.
```

Def. 89. A **Carmichael Number** is a composite n where $a^{n-1} \equiv 1 \pmod{n}$ holds for all a coprime to n (e.g. $561 = 3 \times 11 \times 17$), causing Fermat test to almost always fail.

Def. 90. Miller-Rabin Certificate: Let $n > 2$ be odd. Write $n - 1 = 2^k d$ with d odd. If n is prime and $a \in \{1, \dots, n-1\}$, then either:

$$a^d \equiv 1 \pmod{n}$$

$$\text{or } \exists i \in \{0, \dots, k-1\} : a^{2^i d} \equiv n - 1 \pmod{n}$$

If this fails, a is a certificate proving n is composite.

Thm. 91. Miller-Rabin: If n is prime, output is always correct. If n is composite, error $\leq \frac{1}{4}$ (even for Carmichael numbers). Repeating k times gives error $\leq 4^{-k}$.

6.3 Sorting and Selecting

Thm. 92. The expected runtime of randomized quicksort is:

$$\mathbb{E}[T_{1,n}] \leq 2(n+1)\ln(n) + O(n)$$

Thm. 93. The expected runtime of randomized quickselect is $O(n)$, specifically $\leq 8n$ comparisons. Key insight: with prob $\geq 1/2$ the pivot reduces the problem size by factor $3/4$.

6.4 Duplicate Detection

Algorithm 94

```
#Duplicate Detection via Hashing
1. Hash all elements: X = [(hash(Ai), i)]
2. Sort by hash value
3. For each block of equal hash values:
Compare actual data for all pairs in block
#Expected data comparisons: O(#duplicates) +
      O(1).
```

6.5 Floyd-Cycle Detection

Def. 95. Given $A[1, \dots, n]$ with $A[i] \in \{1, \dots, n\}$, find $i \neq j$ with $A[i] = A[j]$. Model as directed graph $i \to A[i]$.

Lem. 96. If there are duplicates, $\exists t \leq n$ such that $X_t = X_{2t}$ (tortoise-hare). If $X_t = X_{2t}$ then $X_{t+k} = X_k$ for all $k \geq 0$.

Algorithm 97

```
#Floyd-Cycle Detection O(n) time, O(1) space
tortoise = A[n]; hare = A[A[n]]
1. Advance tortoise by 1, hare by 2 until
      they meet
2. Reset hare to n, advance both by 1 until
      they meet
Return the two indices from the last step
```

6.6 Long-Path Problem

Def. 98. Given $G = (V, E)$ and $B \in \mathbb{N}$, decide if $\exists$ a path of length B in G.

Thm. 99. If long-path is solvable in $t(n)$, then Hamiltonian decision is solvable in $t(2n-2) + O(n^2)$.

Def. 100. A **colorful path** in a k-edge-colored graph is a path where every edge has a different color. Define $P_i(v) := \{S \subset [k] \mid |S| = i+1, \exists$ colorful path from v using colors $S\}$.

Lem. 101. DP: $P_0(v) = \{\{c(v)\}\}$ and

$$P_i(v) = \bigcup_{u \in N(v)} \{S \cup \{c(v)\} \mid S \in P_{i-1}(u) \wedge c(v) \notin S\}$$

Thm. 102. Runtime of colorful path DP: $O(2^k \cdot k \cdot m)$. For a path of length $k-1$:

$$P[\exists \text{ colorful path}] \geq \frac{k!}{k^k} \geq e^{-k}$$

Repeated λe^k times: runtime $O(\lambda \cdot (2e)^k \cdot k \cdot m)$, error $e^{-\lambda}$.

6.7 Minimum Cut in Multigraphs

Def. 103. A **multigraph** allows multiple edges between two vertices. An **Edge Cut** $C \subseteq E$ makes $G \setminus C$ disconnected. The minimum edge cut has cardinality $\mu(G)$.

Lem. 104. $2|E| = \sum_v \deg(v)$ and $\mu(G) \leq \min_v \deg(v)$.

Def. 105. Edge Contraction on (u, v): merge u, v into w, redirect all incident edges to w.

Thm. 106. Contract $n-2$ random edges to get 2 vertices. The min cut is preserved with probability $\geq \binom{n}{2}^{-1}$. Repeating $\lambda\binom{n}{2}$ times gives success probability $1 - e^{-\lambda}$.

Thm. 107. Using maxflow: global min cut in $O(n^4 \log n)$ by fixing s and iterating over all t.

6.7.1 Bootstrapping

Thm. 108. Stop contraction at $|G| = t$, compute rest exactly. With $t = \sqrt{n}$: total runtime $O(\lambda n^3)$, error $e^{-\lambda}$.

6.8 Smallest Enclosing Disk

Def. 109. Given $P \subset \mathbb{R}^2$, find the smallest disk C_P containing P.

Lem. 110. $\exists$ a certificate $C(P)$ of ≤ 3 points defining C_P.

Thm. 111. Randomized algorithm: sample Q of 3 points, check $P \subseteq C(Q)$, double copies of points outside $C(Q)$. Using sampling lemma with $r = 11$:

$$\mathbb{E}[|P' \setminus C^\bullet(R)|] \leq 3\frac{N}{r+1}$$

Expected runtime: $O(n \log n)$.

6.9 Convex Hull

Def. 112. The **Convex Hull** $\mathcal{CH}(P)$ is the intersection of all convex sets containing P:

$$\mathcal{CH}(P) = \left\{\sum_i \lambda_i p_i \;\middle|\; \lambda_i \geq 0, \sum \lambda_i = 1\right\}$$

Def. 113. For points p, q, r, the **CCW** test:

$$\text{CCW}(p, q, r) = (x_q - x_p)(y_r - y_p) - (y_q - y_p)(x_r - x_p)$$

> 0: left (CCW), < 0: right (CW), $= 0$: collinear.

Thm. 114. Gift Wrapping (Jarvis March): Iteratively find the next border edge by scanning all points for the smallest angle. Runtime $O(nh)$ where $h =$ number of hull vertices.

6.10 Reachable Vertices

Def. 115. Given directed $G = (V, E)$, for each $v \in V$, compute $n_v = |R(v)|$ (number of reachable vertices).

> **Rem. 116.** Exact: $O(n(n+m))$. Subquadratic likely contradicts SETH.

Thm. 117. Assign each vertex $r_v \sim \text{Uniform}([0, 1])$. Compute $x_v = \min_{u \in R(v)} r_u$ via reverse DFS. Repeat $l = \lceil 2\log_2(2n/\delta)\rceil$ times, estimate $\tilde{n}_v$ as reciprocal of median. Runtime: $O((n \log n + m)\log(n/\delta))$.

Thm. 118. With probability $\geq 1-\delta$: $n_v/20 \leq \tilde{n}_v \leq 20n_v$ for all v. Can be improved to $(1 \pm \epsilon)$-approximation with $O(\frac{1}{\epsilon^2}\log(\frac{n}{\delta}))$ iterations.

7. Flow Networks

Def. 119. A **Flow Network** is a directed graph $G = (V, E)$ with source s, sink t, and capacity $c : E \to \mathbb{R}_{\geq 0}$.

Def. 120. A **flow** $f : E \to \mathbb{R}_{\geq 0}$ satisfies:

- **Capacity**: $0 \leq f(e) \leq c(e)$ for all $e \in E$.
- **Conservation**: $\sum_{e \text{ in}} f(e) = \sum_{e \text{ out}} f(e)$ for all $v \neq s, t$.

The **value** $|f| := \sum_{e \text{ out } s} f(e) - \sum_{e \text{ in } s} f(e)$.

7.1 Cuts

Def. 121. An s-t **cut** (S, T) partitions V with $s \in S, t \in T$. Its capacity:

$$c(S, T) := \sum_{\substack{e=(u,v)\\ u \in S, v \in T}} c(e)$$

Lem. 122. For any flow f and s-t cut (S, T):

$$|f| = \sum_{e \in (S,T)} f(e) - \sum_{e \in (T,S)} f(e)$$

Thm. 123. For any flow f and any s-t cut (S, T): $|f| \leq c(S, T)$.

7.2 Max-Flow Min-Cut

Thm. 124. Max-Flow Min-Cut Theorem: $\max_f |f| = \min_{(S,T)} c(S, T)$. The following are equivalent: (1) f is a max flow. (2) No augmenting path in G_f. (3) $\exists$ cut (S, T) with $|f| = c(S, T)$.

7.3 Residual Networks

Def. 125. The **residual network** $G_f = (V, E_f)$ has:

$$c_f(u, v) = \begin{cases} c(u, v) - f(u, v) & (u, v) \in E \\ f(v, u) & (v, u) \in E \\ 0 & \text{otherwise} \end{cases}$$

$E_f = \{(u, v) \mid c_f(u, v) > 0\}$. Back edges allow flow rerouting.

Def. 126. An **augmenting path** is a directed s-t path in G_f. Its **bottleneck** is $\Delta(f, P) := \min_{e \in P} c_f(e)$.

Lem. 127. Augmenting f along P (add Δ on forward, subtract on backward edges) yields valid flow f' with $|f'| = |f| + \Delta$.

7.4 Ford-Fulkerson

Algorithm 128

```
#Ford-Fulkerson Algorithm
Init f(e) = 0 for all e
While exists augmenting path P in G_f:
delta = min residual capacity on P
Update f along P
Return f
```

Thm. 129. For integer capacities, Ford-Fulkerson terminates in $O(|f^*| \cdot |E|)$ time.

Imp. 130. Ford-Fulkerson is **not guaranteed to terminate** for irrational capacities. For rational capacities it terminates but can be exponentially slow.

7.5 Applications of MaxFlow

7.5.1 Maximum Bipartite Matching

> **Rem. 131.** Construct flow network: add $s \to A$, $B \to t$ with capacity 1, and edges from E with capacity 1. Max integer flow = size of max matching. Solvable in $O(mn)$.

7.5.2 Edge-disjoint Paths

Def. 132. Given s, t, find max set of edge-disjoint s-t paths.

Thm. 133. Replace each undirected edge with two directed edges of capacity 1. Then the max number of edge-disjoint paths equals $\text{maxFlow}(N_G) = \text{minCut}(S, T)$ (Menger's Theorem).

7.5.3 Image Segmentation

Def. 134. Given pixels P with foreground/background preferences α_p, β_p and neighbor confidence γ_e. Find partition (A, B) maximizing:

$$q(A, B) = \sum_{p \in A} \alpha_p + \sum_{p \in B} \beta_p - \sum_{e \in E'} \gamma_e$$

Thm. 135. Create flow network: edges $s \to p$ with capacity α_p, $p \to t$ with capacity β_p, and bidirectional edges between neighbors with capacity γ_e. The min-cut corresponds to the maximum quality partition.