Randomised Algorithms

Lecture 1: Introduction

Thomas Sauerwald (tms41@cam.ac.uk)

Lent 2022

UNIVERSITY OF CAMBRIDGE

Outline

Randomised Algorithms

What? Randomised Algorithms utilise random bits to compute their output.

Randomised Algorithms

What? Randomised Algorithms utilise random bits to compute their output.

Why? Randomised Algorithms often provide an efficient (and elegant!) solution or approximation to a problem that is costly (or impossible) to solve deterministically.

Randomised Algorithms

What? Randomised Algorithms utilise random bits to compute their output.

Why? Randomised Algorithms often provide an efficient (and elegant!) solution or approximation to a problem that is costly (or impossible) to solve deterministically.

But sometimes: simple algorithm at the cost of a complicated analysis!

Randomised Algorithms

What? Randomised Algorithms utilise random bits to compute their output.

Why? Randomised Algorithms often provide an efficient (and elegant!) solution or approximation to a problem that is costly (or impossible) to solve deterministically.

"... If somebody would ask me, what in the last 10 years, what was the most important change in the study of algorithms I would have to say that people getting really familiar with randomised algorithms had to be the winner."
\- *Donald E. Knuth* (in *Randomization and Religion*)

Randomised Algorithms

What? Randomised Algorithms utilise random bits to compute their output.

Why? Randomised Algorithms often provide an efficient (and elegant!) solution or approximation to a problem that is costly (or impossible) to solve deterministically.

"... If somebody would ask me, what in the last 10 years, what was the most important change in the study of algorithms I would have to say that people getting really familiar with randomised algorithms had to be the winner."
\- *Donald E. Knuth* (in *Randomization and Religion*)

How? This course aims to strengthen your knowledge of probability theory and apply this to analyse examples of randomised algorithms.

Randomised Algorithms

What? Randomised Algorithms utilise random bits to compute their output.

Why? Randomised Algorithms often provide an efficient (and elegant!) solution or approximation to a problem that is costly (or impossible) to solve deterministically.

"... If somebody would ask me, what in the last 10 years, what was the most important change in the study of algorithms I would have to say that people getting really familiar with randomised algorithms had to be the winner."
\- *Donald E. Knuth* (in *Randomization and Religion*)

How? This course aims to strengthen your knowledge of probability theory and apply this to analyse examples of randomised algorithms.

What if I (initially) don't care about randomised algorithms?
Many of the techniques in this course (Markov Chains, Concentration of Measure, Spectral Theory) are very relevant to other popular areas of research and employment such as Data Science and Machine Learning.

Some stuff you should know...

In this course we will assume some basic knowledge of probability:

- random variable
- computing expectations and variances
- notions of independence
- “general” idea of how to compute probabilities (manipulating, counting and **estimating**)

Some stuff you should know...

In this course we will assume some basic knowledge of probability:

- random variable
- computing expectations and variances
- notions of independence
- "general" idea of how to compute probabilities (manipulating, counting and **estimating**)

You should also be familiar with basic computer science, mathematics knowledge such as:

- graphs
- basic algorithms (sorting, graph algorithms etc.)
- matrices, norms and vectors

Outline

1 Introduction (Lecture)

- Intro to Randomised Algorithms; Logistics(?); Recap of Probability; Examples.

1 **Introduction** (Lecture)

- Intro to Randomised Algorithms; Logistics(?); Recap of Probability; Examples.

Lectures 2-5 focus on probabilistic tools and techniques.

2–3 **Concentration** (Lectures)

- Concept of Concentration; Recap of Markov and Chebyshev; Chernoff Bounds and Applications; Extensions: Hoeffding's Inequality and Method of Bounded Differences; Applications.

4 **Markov Chains and Mixing Times** (Lecture)

- Recap; Stopping and Hitting Times; Properties of Markov Chains; Convergence to Stationary Distribution; Variation Distance and Mixing Time

5 **Hitting Times and Application to 2-SAT** (Lecture)

- Reversible Markov Chains and Random Walks on Graphs; Cover Times and Hitting Times on Graphs (Example: Paths and Grids); A Randomised Algorithm for 2-SAT Algorithm

1 Introduction (Lecture)

- Intro to Randomised Algorithms; Logistics(?); Recap of Probability; Examples.

Lectures 2-5 focus on probabilistic tools and techniques.

2–3 Concentration (Lectures)

- Concept of Concentration; Recap of Markov and Chebyshev; Chernoff Bounds and Applications; Extensions: Hoeffding's Inequality and Method of Bounded Differences; Applications.

4 Markov Chains and Mixing Times (Lecture)

- Recap; Stopping and Hitting Times; Properties of Markov Chains; Convergence to Stationary Distribution; Variation Distance and Mixing Time

5 Hitting Times and Application to 2-SAT (Lecture)

- Reversible Markov Chains and Random Walks on Graphs; Cover Times and Hitting Times on Graphs (Example: Paths and Grids); A Randomised Algorithm for 2-SAT Algorithm

Lectures 6-8 introduce linear programming, a (mostly) deterministic but very powerful technique to solve various optimisation problems.

6–7 Linear Programming (Lectures)

- Introduction to Linear Programming, Applications, Standard and Slack Forms, Simplex Algorithm, Finding an Initial Solution, Fundamental Theorem of Linear Programming

8 Travelling Salesman Problem (Interactive Demo)

- Hardness of the general TSP problem, Formulating TSP as an integer program; Classical TSP instance from 1954; Branch & Bound Technique to solve integer programs using linear programs

We then see how we can efficiently combine linear programming with randomised techniques, in particular, rounding:

9–10 **Randomised Approximation Algorithms** (Lectures)

- MAX-3-CNF and Guessing, Vertex-Cover and Deterministic Rounding of Linear Program, Set-Cover and Randomised Rounding, Concluding Example: MAX-CNF and Hybrid Algorithm

We then see how we can efficiently combine linear programming with randomised techniques, in particular, rounding:

9–10 **Randomised Approximation Algorithms** (Lectures)

- MAX-3-CNF and Guessing, Vertex-Cover and Deterministic Rounding of Linear Program, Set-Cover and Randomised Rounding, Concluding Example: MAX-CNF and Hybrid Algorithm

Lectures 11-16 cover more advanced topics with a ML flavour:

11–12 **Spectral Graph Theory and Spectral Clustering** (Lectures)

- Eigenvalues, Eigenvectors and Spectrum; Visualising Graphs; Expansion; Cheeger's Inequality; Clustering and Examples; Analysing Mixing Times

13 **Streaming Algorithms** (Lecture)

- Motivation and Concepts of Algorithms for Data Streams; Approximate Counting and Morris Algorithm; Hash Functions; Approximating Frequency Moments

14 **Online Learning with Experts** (Lecture)

- Online and Reinforcement Learning Framework; Weighted Majority Algorithm and Analysis; Randomised Weighted Majority Algorithm; Learning Rate

15–16 **Algorithms for Multi-Armed Bandits** (Lectures)

- Definition and Types of Bandit Problems; Regret in Stochastic Bandits; Algorithms: Greedy, Epsilon-Greedy and UCB; A Special Instance with Two Bandits; Adversarial Bandits: EXP3 and Connection to Weighted Majority.

Outline

Introduction

Topics and Syllabus

A (Very) Brief Reminder of Probability Theory and Examples

Recap: Probability Space

In probability theory we wish to evaluate the likelihood of certain results from an experiment. The setting of this is the *probability space* $(\Omega, \Sigma, \mathbf{P})$.

Components of the Probability Space $(\Omega, \Sigma, \mathbf{P})$

- The *Sample Space* Ω contains all the possible *outcomes* $\omega_1, \omega_2, \ldots$ of the experiment.
- The *Event Space* Σ is the power-set of Ω containing *events*, which are combinations of outcomes (subsets of Ω including $\emptyset$ and Ω).
- The *Probability Measure* $\mathbf{P}$ is a function from Σ to $\mathbb{R}$ satisfying
  (i) $0 \leq \mathbf{P}[\,\mathcal{E}\,] \leq 1$, for all $\mathcal{E} \in \Sigma$
  (ii) $\mathbf{P}[\,\Omega\,] = 1$
  (iii) If $\mathcal{E}_1, \mathcal{E}_2, \ldots \in \Sigma$ are pairwise disjoint ($\mathcal{E}_i \cap \mathcal{E}_j = \emptyset$ for all $i \neq j$) then

$$\mathbf{P}\left[\bigcup_{i=1}^{\infty} \mathcal{E}_i\right] = \sum_{i=1}^{\infty} \mathbf{P}[\,\mathcal{E}_i\,].$$

Recap: Random Variables

A *random variable* X on $(\Omega, \Sigma, \mathbf{P})$ is a function $X : \Omega \to \mathbb{R}$ mapping each sample "outcome" to a real number.

Intuitively, random variables are the "observables" in our experiment.

Recap: Random Variables

A *random variable* X on $(\Omega, \Sigma, \mathbf{P})$ is a function $X : \Omega \to \mathbb{R}$ mapping each sample "outcome" to a real number.

Intuitively, random variables are the "observables" in our experiment.

Examples of random variables

- The number of heads in three coin flips $X_1, X_2, X_3 \in \{0, 1\}$ is:

$$X_1 + X_2 + X_3$$

Recap: Random Variables

A *random variable* X on $(\Omega, \Sigma, \mathbf{P})$ is a function $X : \Omega \to \mathbb{R}$ mapping each sample "outcome" to a real number.

Intuitively, random variables are the "observables" in our experiment.

Examples of random variables

- The number of heads in three coin flips $X_1, X_2, X_3 \in \{0, 1\}$ is:

$$X_1 + X_2 + X_3$$

- The indicator random variable $\mathbf{1}_{\mathcal{E}}$ of an event $\mathcal{E} \in \Sigma$ given by

$$\mathbf{1}_{\mathcal{E}}(\omega) = \begin{cases} 1 & \text{if } \omega \in \mathcal{E} \\ 0 & \text{otherwise.} \end{cases}$$

Recap: Random Variables

A *random variable* X on $(\Omega, \Sigma, \mathbf{P})$ is a function $X : \Omega \to \mathbb{R}$ mapping each sample "outcome" to a real number.

Intuitively, random variables are the "observables" in our experiment.

Examples of random variables

- The number of heads in three coin flips $X_1, X_2, X_3 \in \{0, 1\}$ is:

$$X_1 + X_2 + X_3$$

- The indicator random variable $\mathbf{1}_{\mathcal{E}}$ of an event $\mathcal{E} \in \Sigma$ given by

$$\mathbf{1}_{\mathcal{E}}(\omega) = \begin{cases} 1 & \text{if } \omega \in \mathcal{E} \\ 0 & \text{otherwise.} \end{cases}$$

For the indicator random variable $\mathbf{1}_{\mathcal{E}}$ we have $\mathbf{E}[\mathbf{1}_{\mathcal{E}}] = \mathbf{P}[\mathcal{E}]$.

Recap: Random Variables

A *random variable* X on $(\Omega, \Sigma, \mathbf{P})$ is a function $X : \Omega \to \mathbb{R}$ mapping each sample "outcome" to a real number.

Intuitively, random variables are the "observables" in our experiment.

Examples of random variables

- The number of heads in three coin flips $X_1, X_2, X_3 \in \{0, 1\}$ is:

$$X_1 + X_2 + X_3$$

- The indicator random variable $\mathbf{1}_{\mathcal{E}}$ of an event $\mathcal{E} \in \Sigma$ given by

$$\mathbf{1}_{\mathcal{E}}(\omega) = \begin{cases} 1 & \text{if } \omega \in \mathcal{E} \\ 0 & \text{otherwise.} \end{cases}$$

  For the indicator random variable $\mathbf{1}_{\mathcal{E}}$ we have $\mathbf{E}\left[\,\mathbf{1}_{\mathcal{E}}\,\right] = \mathbf{P}\left[\,\mathcal{E}\,\right]$.
- The number of sixes of two dice throws $X_1, X_2 \in \{1, 2, \ldots, 6\}$ is

$$\mathbf{1}_{X_1=6} + \mathbf{1}_{X_2=6}$$

A Randomised Algorithm for MAX-CUT (1/2)

$E(A, B)$: set of edges with one endpoint in $A \subseteq V$ and the other in $B \subseteq V$.

A Randomised Algorithm for MAX-CUT (1/2)

$E(A, B)$: set of edges with one endpoint in $A \subseteq V$ and the other in $B \subseteq V$.

MAX-CUT Problem

A Randomised Algorithm for MAX-CUT (1/2)

$E(A, B)$: set of edges with one endpoint in $A \subseteq V$ and the other in $B \subseteq V$.

MAX-CUT Problem

- Given: Undirected graph $G = (V, E)$

A Randomised Algorithm for MAX-CUT (1/2)

$E(A, B)$: set of edges with one endpoint in $A \subseteq V$ and the other in $B \subseteq V$.

MAX-CUT Problem

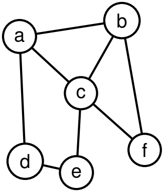
- Given: Undirected graph $G = (V, E)$
- Goal: Find $S \subseteq V$ such that $e(S, S^c) := |E(S, V \setminus S)|$ is maximised.

A Randomised Algorithm for MAX-CUT (1/2)

$E(A, B)$: set of edges with one endpoint in $A \subseteq V$ and the other in $B \subseteq V$.

MAX-CUT Problem

- Given: Undirected graph $G = (V, E)$
- Goal: Find $S \subseteq V$ such that $e(S, S^c) := |E(S, V \setminus S)|$ is maximised.

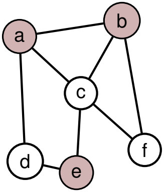


A Randomised Algorithm for MAX-CUT (1/2)

$E(A, B)$: set of edges with one endpoint in $A \subseteq V$ and the other in $B \subseteq V$.

MAX-CUT Problem

- Given: Undirected graph $G = (V, E)$
- Goal: Find $S \subseteq V$ such that $e(S, S^c) := |E(S, V \setminus S)|$ is maximised.



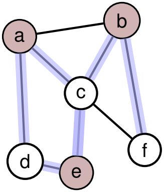
$S = \{a, b, e\}$
$e(S, S^c) = 6$

A Randomised Algorithm for MAX-CUT (1/2)

$E(A, B)$: set of edges with one endpoint in $A \subseteq V$ and the other in $B \subseteq V$.

MAX-CUT Problem

- Given: Undirected graph $G = (V, E)$
- Goal: Find $S \subseteq V$ such that $e(S, S^c) := |E(S, V \setminus S)|$ is maximised.



$S = \{a, b, e\}$
$e(S, S^c) = 6$

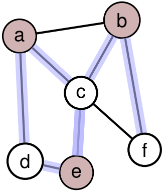
A Randomised Algorithm for MAX-CUT (1/2)

$E(A, B)$: set of edges with one endpoint in $A \subseteq V$ and the other in $B \subseteq V$.

MAX-CUT Problem

- Given: Undirected graph $G = (V, E)$
- Goal: Find $S \subseteq V$ such that $e(S, S^c) := |E(S, V \setminus S)|$ is maximised.

Applications:



$S = \{a, b, e\}$
$e(S, S^c) = 6$

A Randomised Algorithm for MAX-CUT (1/2)

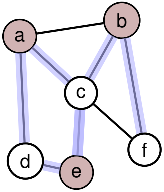
$E(A, B)$: set of edges with one endpoint in $A \subseteq V$ and the other in $B \subseteq V$.

MAX-CUT Problem

- Given: Undirected graph $G = (V, E)$
- Goal: Find $S \subseteq V$ such that $e(S, S^c) := |E(S, V \setminus S)|$ is maximised.

Applications:

- network design, VLSI design



$S = \{a, b, e\}$
$e(S, S^c) = 6$

A Randomised Algorithm for MAX-CUT (1/2)

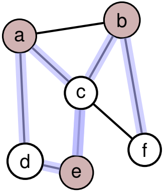
$E(A, B)$: set of edges with one endpoint in $A \subseteq V$ and the other in $B \subseteq V$.

MAX-CUT Problem

- Given: Undirected graph $G = (V, E)$
- Goal: Find $S \subseteq V$ such that $e(S, S^c) := |E(S, V \setminus S)|$ is maximised.

Applications:

- network design, VLSI design
- clustering, statistical physics



$S = \{a, b, e\}$
$e(S, S^c) = 6$

A Randomised Algorithm for MAX-CUT (1/2)

$E(A, B)$: set of edges with one endpoint in $A \subseteq V$ and the other in $B \subseteq V$.

MAX-CUT Problem

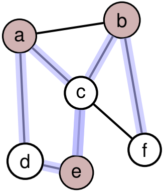
- Given: Undirected graph $G = (V, E)$
- Goal: Find $S \subseteq V$ such that $e(S, S^c) := |E(S, V \setminus S)|$ is maximised.

Applications:

- network design, VLSI design
- clustering, statistical physics

Comments:

- This example will appear again in the course



$S = \{a, b, e\}$
$e(S, S^c) = 6$

A Randomised Algorithm for MAX-CUT (1/2)

$E(A, B)$: set of edges with one endpoint in $A \subseteq V$ and the other in $B \subseteq V$.

MAX-CUT Problem

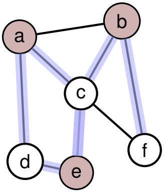
- Given: Undirected graph $G = (V, E)$
- Goal: Find $S \subseteq V$ such that $e(S, S^c) := |E(S, V \setminus S)|$ is maximised.

Applications:

- network design, VLSI design
- clustering, statistical physics

Comments:

- This example will appear again in the course
- MAX-CUT is NP-hard



$S = \{a, b, e\}$
$e(S, S^c) = 6$

A Randomised Algorithm for MAX-CUT (1/2)

$E(A, B)$: set of edges with one endpoint in $A \subseteq V$ and the other in $B \subseteq V$.

MAX-CUT Problem

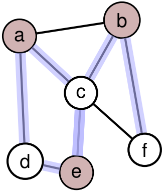
- Given: Undirected graph $G = (V, E)$
- Goal: Find $S \subseteq V$ such that $e(S, S^c) := |E(S, V \setminus S)|$ is maximised.

Applications:

- network design, VLSI design
- clustering, statistical physics

Comments:

- This example will appear again in the course
- MAX-CUT is NP-hard
- It is different from the clustering problem, where we want to find a sparse cut



$S = \{a, b, e\}$
$e(S, S^c) = 6$

A Randomised Algorithm for MAX-CUT (1/2)

$E(A, B)$: set of edges with one endpoint in $A \subseteq V$ and the other in $B \subseteq V$.

MAX-CUT Problem

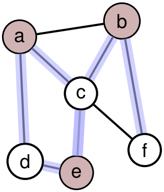
- Given: Undirected graph $G = (V, E)$
- Goal: Find $S \subseteq V$ such that $e(S, S^c) := |E(S, V \setminus S)|$ is maximised.

Applications:

- network design, VLSI design
- clustering, statistical physics

Comments:

- This example will appear again in the course
- MAX-CUT is NP-hard
- It is different from the clustering problem, where we want to find a sparse cut
- Note that the MIN-CUT problem is solvable in polynomial time!



$S = \{a, b, e\}$
$e(S, S^c) = 6$

A Randomised Algorithm for MAX-CUT (2/2)

RANDMAXCUT (Input $G = (V, E)$)

Start with $S \leftarrow \emptyset$

For each $v \in V$, add v to S with probability $1/2$

Return S

A Randomised Algorithm for MAX-CUT (2/2)

RandMaxCut (Input $G = (V, E)$)
1: Start with $S \leftarrow \emptyset$
2: **For** each $v \in V$, add v to S with probability $1/2$
3: **Return** S

Proposition

RandMaxCut gives a 2-approximation using time $O(n)$.

A Randomised Algorithm for MAX-CUT (2/2)

RandMaxCut (Input $G = (V, E)$)

Start with $S \leftarrow \emptyset$

For each $v \in V$, add v to S with probability $1/2$

Return S

Proposition

RandMaxCut gives a 2-approximation using time $O(n)$.

Exercise: What is the sample space Ω and event space Σ here? Which random variable do we need to analyse?

A Randomised Algorithm for MAX-CUT (2/2)

RandMaxCut (Input $G = (V, E)$)

1: Start with $S \leftarrow \emptyset$
2: **For** each $v \in V$, add v to S with probability $1/2$
3: **Return** S

Proposition

RandMaxCut gives a 2-approximation using time $O(n)$.

Proof:

A Randomised Algorithm for MAX-CUT (2/2)

RandMaxCut (Input $G = (V, E)$)

1: Start with $S \leftarrow \emptyset$
2: **For** each $v \in V$, add v to S with probability $1/2$
3: **Return** S

Proposition

RandMaxCut gives a 2-approximation using time $O(n)$.

Proof:

- We need to analyse the expectation of $e(S, S^c)$:

$$\mathbf{E}\left[\, e\left(S, S^c\right) \,\right]$$

A Randomised Algorithm for MAX-CUT (2/2)

RANDMAXCUT (Input $G = (V, E)$)

1: Start with $S \leftarrow \emptyset$
2: **For** each $v \in V$, add v to S with probability $1/2$
3: **Return** S

Proposition

RandMaxCut gives a 2-approximation using time $O(n)$.

Proof:

- We need to analyse the expectation of $e(S, S^c)$:

$$\mathbf{E}\left[\, e(S, S^c) \,\right] = \mathbf{E}\left[\sum_{\{u,v\} \in E} \mathbf{1}_{\{u \in S, v \in S^c\} \cup \{u \in S^c, v \in S\}} \right]$$

A Randomised Algorithm for MAX-CUT (2/2)

RandMaxCut (Input $G = (V, E)$)

1: Start with $S \leftarrow \emptyset$
2: **For** each $v \in V$, add v to S with probability $1/2$
3: **Return** S

Proposition

RandMaxCut gives a 2-approximation using time $O(n)$.

Proof:

- We need to analyse the expectation of $e(S, S^c)$:

$$\mathbf{E}\left[\, e(S, S^c) \,\right] = \mathbf{E}\left[\sum_{\{u,v\} \in E} \mathbf{1}_{\{u \in S, v \in S^c\} \cup \{u \in S^c, v \in S\}} \right]$$

$$= \sum_{\{u,v\} \in E} \mathbf{E}\left[\, \mathbf{1}_{\{u \in S, v \in S^c\} \cup \{u \in S^c, v \in S\}} \,\right]$$

A Randomised Algorithm for MAX-CUT (2/2)

RANDMAXCUT (Input $G = (V, E)$)

1: Start with $S \leftarrow \emptyset$
2: **For** each $v \in V$, add v to S with probability $1/2$
3: **Return** S

Proposition

RandMaxCut gives a 2-approximation using time $O(n)$.

Proof:

- We need to analyse the expectation of $e(S, S^c)$:

$$\begin{aligned}
\mathbf{E}\left[\, e(S, S^c) \,\right] &= \mathbf{E}\left[\sum_{\{u,v\} \in E} \mathbf{1}_{\{u \in S, v \in S^c\} \cup \{u \in S^c, v \in S\}} \right] \\
&= \sum_{\{u,v\} \in E} \mathbf{E}\left[\, \mathbf{1}_{\{u \in S, v \in S^c\} \cup \{u \in S^c, v \in S\}} \,\right] \\
&= \sum_{\{u,v\} \in E} \mathbf{P}\left[\, \{u \in S, v \in S^c\} \cup \{u \in S^c, v \in S\} \,\right]
\end{aligned}$$

A Randomised Algorithm for MAX-CUT (2/2)

RANDMAXCUT (Input $G = (V, E)$)
1: Start with $S \leftarrow \emptyset$
2: **For** each $v \in V$, add v to S with probability $1/2$
3: **Return** S

Proposition

RandMaxCut gives a 2-approximation using time $O(n)$.

Proof:

- We need to analyse the expectation of $e(S, S^c)$:

$$\begin{aligned}
\mathbf{E}\left[\, e(S, S^c) \,\right] &= \mathbf{E}\left[\sum_{\{u,v\} \in E} \mathbf{1}_{\{u \in S, v \in S^c\} \cup \{u \in S^c, v \in S\}} \right] \\
&= \sum_{\{u,v\} \in E} \mathbf{E}\left[\mathbf{1}_{\{u \in S, v \in S^c\} \cup \{u \in S^c, v \in S\}} \right] \\
&= \sum_{\{u,v\} \in E} \mathbf{P}\left[\{u \in S, v \in S^c\} \cup \{u \in S^c, v \in S\} \right] \\
&= 2 \sum_{\{u,v\} \in E} \mathbf{P}\left[u \in S, v \in S^c \right]
\end{aligned}$$

A Randomised Algorithm for MAX-CUT (2/2)

RANDMAXCUT (Input $G = (V, E)$)

1: Start with $S \leftarrow \emptyset$

2: **For** each $v \in V$, add v to S with probability $1/2$

3: **Return** S

Proposition

RandMaxCut gives a 2-approximation using time $O(n)$.

Proof:

- We need to analyse the expectation of $e(S, S^c)$:

$$\begin{aligned}
\mathbf{E}\left[\, e(S, S^c) \,\right] &= \mathbf{E}\left[\sum_{\{u,v\}\in E} \mathbf{1}_{\{u\in S, v\in S^c\}\cup\{u\in S^c, v\in S\}} \right] \\
&= \sum_{\{u,v\}\in E} \mathbf{E}\left[\, \mathbf{1}_{\{u\in S, v\in S^c\}\cup\{u\in S^c, v\in S\}} \,\right] \\
&= \sum_{\{u,v\}\in E} \mathbf{P}\left[\, \{u \in S, v \in S^c\} \cup \{u \in S^c, v \in S\} \,\right] \\
&= 2 \sum_{\{u,v\}\in E} \mathbf{P}\left[\, u \in S, v \in S^c \,\right] = 2 \sum_{\{u,v\}\in E} \mathbf{P}\left[\, u \in S \,\right] \cdot \mathbf{P}\left[\, v \in S^c \,\right]
\end{aligned}$$

A Randomised Algorithm for MAX-CUT (2/2)

RANDMAXCUT (Input $G = (V, E)$)

1: Start with $S \leftarrow \emptyset$
2: **For** each $v \in V$, add v to S with probability $1/2$
3: **Return** S

Proposition

RandMaxCut gives a 2-approximation using time $O(n)$.

Proof:

- We need to analyse the expectation of $e(S, S^c)$:

$$\begin{aligned}
\mathbf{E}\left[\, e(S, S^c) \,\right] &= \mathbf{E}\left[\sum_{\{u,v\}\in E} \mathbf{1}_{\{u\in S, v\in S^c\}\cup\{u\in S^c, v\in S\}} \right] \\
&= \sum_{\{u,v\}\in E} \mathbf{E}\left[\, \mathbf{1}_{\{u\in S, v\in S^c\}\cup\{u\in S^c, v\in S\}} \,\right] \\
&= \sum_{\{u,v\}\in E} \mathbf{P}\left[\, \{u \in S, v \in S^c\} \cup \{u \in S^c, v \in S\} \,\right] \\
&= 2 \sum_{\{u,v\}\in E} \mathbf{P}\left[\, u \in S, v \in S^c \,\right] = 2 \sum_{\{u,v\}\in E} \mathbf{P}\left[\, u \in S \,\right] \cdot \mathbf{P}\left[\, v \in S^c \,\right] = |E|/2.
\end{aligned}$$

A Randomised Algorithm for MAX-CUT (2/2)

RANDMAXCUT (Input $G = (V, E)$)

1: Start with $S \leftarrow \emptyset$

2: **For** each $v \in V$, add v to S with probability $1/2$

3: **Return** S

Proposition

RandMaxCut gives a 2-approximation using time $O(n)$.

Proof:

- We need to analyse the expectation of $e(S, S^c)$:

$$\begin{aligned}
\mathbf{E}\left[\, e(S, S^c) \,\right] &= \mathbf{E}\left[\sum_{\{u,v\}\in E} \mathbf{1}_{\{u\in S, v\in S^c\}\cup\{u\in S^c, v\in S\}} \right] \\
&= \sum_{\{u,v\}\in E} \mathbf{E}\left[\, \mathbf{1}_{\{u\in S, v\in S^c\}\cup\{u\in S^c, v\in S\}} \,\right] \\
&= \sum_{\{u,v\}\in E} \mathbf{P}\left[\, \{u \in S, v \in S^c\} \cup \{u \in S^c, v \in S\} \,\right] \\
&= 2 \sum_{\{u,v\}\in E} \mathbf{P}\left[\, u \in S, v \in S^c \,\right] = 2 \sum_{\{u,v\}\in E} \mathbf{P}\left[\, u \in S \,\right] \cdot \mathbf{P}\left[\, v \in S^c \,\right] = |E|/2.
\end{aligned}$$

- Since for any $S \subseteq V$, we have $e(S, S^c) \leq |E|$, concluding the proof.

A Randomised Algorithm for MAX-CUT (2/2)

RandMaxCut (

This kind of "random guessing" will appear a few times in this course!

Start with $S \leftarrow \emptyset$
For each $v \in V$, add v to S with probability $1/2$
Return S

Proposition

RandMaxCut gives a 2-approximation using time $O(n)$.

Proof:

- We need to analyse the expectation of $e(S, S^c)$:

$$
\begin{aligned}
\mathbf{E}\left[\, e(S, S^c) \,\right] &= \mathbf{E}\left[\sum_{\{u,v\} \in E} \mathbf{1}_{\{u \in S, v \in S^c\} \cup \{u \in S^c, v \in S\}} \right] \\
&= \sum_{\{u,v\} \in E} \mathbf{E}\left[\, \mathbf{1}_{\{u \in S, v \in S^c\} \cup \{u \in S^c, v \in S\}} \,\right] \\
&= \sum_{\{u,v\} \in E} \mathbf{P}\left[\, \{u \in S, v \in S^c\} \cup \{u \in S^c, v \in S\} \,\right] \\
&= 2 \sum_{\{u,v\} \in E} \mathbf{P}\left[\, u \in S, v \in S^c \,\right] = 2 \sum_{\{u,v\} \in E} \mathbf{P}\left[\, u \in S \,\right] \cdot \mathbf{P}\left[\, v \in S^c \,\right] = |E|/2.
\end{aligned}
$$

- Since for any $S \subseteq V$, we have $e(S, S^c) \leq |E|$, concluding the proof.

A Randomised Algorithm for MAX-CUT (2/2)

RandMaxCut (

This kind of "random guessing" will appear a few times in this course!

1: Start with $S \leftarrow \emptyset$
2: **For** each $v \in V$, add v to S with probability $1/2$
3: **Return** S

Proposition

RandMaxCut gives a 2-approximation using time $O(n)$.

Later: learn stronger tools that imply concentration around the expectation!

Proof:

- We need to analyse the expectation of $e(S, S^c)$:

$$\begin{aligned}
\mathbf{E}\left[\, e(S, S^c) \,\right] &= \mathbf{E}\left[\sum_{\{u,v\}\in E} \mathbf{1}_{\{u\in S, v\in S^c\}\cup\{u\in S^c, v\in S\}} \right] \\
&= \sum_{\{u,v\}\in E} \mathbf{E}\left[\, \mathbf{1}_{\{u\in S, v\in S^c\}\cup\{u\in S^c, v\in S\}} \,\right] \\
&= \sum_{\{u,v\}\in E} \mathbf{P}\left[\, \{u \in S, v \in S^c\} \cup \{u \in S^c, v \in S\} \,\right] \\
&= 2 \sum_{\{u,v\}\in E} \mathbf{P}\left[\, u \in S, v \in S^c \,\right] = 2 \sum_{\{u,v\}\in E} \mathbf{P}\left[\, u \in S \,\right] \cdot \mathbf{P}\left[\, v \in S^c \,\right] = |E|/2.
\end{aligned}$$

- Since for any $S \subseteq V$, we have $e(S, S^c) \leq |E|$, concluding the proof.

A Randomised Algorithm for MAX-CUT (2/2)

RandMaxCut (

This kind of "random guessing" will appear a few times in this course!

Start with $S \leftarrow \emptyset$
For each $v \in V$, add v to S with probability $1/2$
Return S

More details on approximation algorithms from Lecture 9 onwards!

Proposition

RandMaxCut gives a 2-approximation using time $O(n)$.

Later: learn stronger tools that imply concentration around the expectation!

Proof:

- We need to analyse the expectation of $e(S, S^c)$:

$$\begin{aligned}
\mathbf{E}\left[\, e(S, S^c)\,\right] &= \mathbf{E}\left[\sum_{\{u,v\}\in E} \mathbf{1}_{\{u\in S, v\in S^c\}\cup\{u\in S^c, v\in S\}} \right] \\
&= \sum_{\{u,v\}\in E} \mathbf{E}\left[\, \mathbf{1}_{\{u\in S, v\in S^c\}\cup\{u\in S^c, v\in S\}}\,\right] \\
&= \sum_{\{u,v\}\in E} \mathbf{P}\left[\, \{u\in S, v\in S^c\}\cup\{u\in S^c, v\in S\}\,\right] \\
&= 2 \sum_{\{u,v\}\in E} \mathbf{P}\left[\, u\in S, v\in S^c\,\right] = 2 \sum_{\{u,v\}\in E} \mathbf{P}\left[\, u\in S\,\right]\cdot \mathbf{P}\left[\, v\in S^c\,\right] = |E|/2.
\end{aligned}$$

- Since for any $S \subseteq V$, we have $e(S, S^c) \leq |E|$, concluding the proof.

Boole's Inequality (Union Bound)

Union Bound

Let $\mathcal{E}_1, \ldots, \mathcal{E}_n$ be a collection of events in Σ. Then

$$\mathbf{P}\left[\bigcup_{i=1}^{n} \mathcal{E}_i\right] \leq \sum_{i=1}^{n} \mathbf{P}\left[\,\mathcal{E}_i\,\right].$$

Boole's Inequality (Union Bound)

Union Bound is one of the most basic probability inequalities, yet it is extremely useful and easy to apply!

Union Bound

Let $\mathcal{E}_1, \ldots, \mathcal{E}_n$ be a collection of events in Σ. Then

$$\mathbf{P}\left[\bigcup_{i=1}^{n} \mathcal{E}_i\right] \leq \sum_{i=1}^{n} \mathbf{P}\left[\,\mathcal{E}_i\,\right].$$

Boole's Inequality (Union Bound)

Union Bound is one of the most basic probability inequalities, yet it is extremely useful and easy to apply!

Union Bound

Let $\mathcal{E}_1, \ldots, \mathcal{E}_n$ be a collection of events in Σ. Then

$$\mathbf{P}\left[\bigcup_{i=1}^{n} \mathcal{E}_i\right] \leq \sum_{i=1}^{n} \mathbf{P}[\,\mathcal{E}_i\,].$$

A Proof using Indicator Random Variables:

Boole's Inequality (Union Bound)

Union Bound is one of the most basic probability inequalities, yet it is extremely useful and easy to apply!

Union Bound

Let $\mathcal{E}_1, \ldots, \mathcal{E}_n$ be a collection of events in Σ. Then

$$\mathbf{P}\left[\bigcup_{i=1}^{n} \mathcal{E}_i\right] \leq \sum_{i=1}^{n} \mathbf{P}\left[\mathcal{E}_i\right].$$

A Proof using Indicator Random Variables:

1. Let $\mathbf{1}_{\mathcal{E}_i}$ be the random variable that takes value 1 if $\mathcal{E}_i$ holds, 0 otherwise

Boole's Inequality (Union Bound)

Union Bound is one of the most basic probability inequalities, yet it is extremely useful and easy to apply!

Union Bound

Let $\mathcal{E}_1, \ldots, \mathcal{E}_n$ be a collection of events in Σ. Then

$$\mathbf{P}\left[\bigcup_{i=1}^{n} \mathcal{E}_i\right] \leq \sum_{i=1}^{n} \mathbf{P}\left[\mathcal{E}_i\right].$$

A Proof using Indicator Random Variables:

1. Let $\mathbf{1}_{\mathcal{E}_i}$ be the random variable that takes value 1 if $\mathcal{E}_i$ holds, 0 otherwise
2. $\mathbf{E}\left[\mathbf{1}_{\mathcal{E}_i}\right] = \mathbf{P}\left[\mathcal{E}_i\right]$ (**Check this**)

Boole's Inequality (Union Bound)

Union Bound is one of the most basic probability inequalities, yet it is extremely useful and easy to apply!

Union Bound

Let $\mathcal{E}_1, \ldots, \mathcal{E}_n$ be a collection of events in Σ. Then

$$\mathbf{P}\left[\bigcup_{i=1}^{n} \mathcal{E}_i\right] \leq \sum_{i=1}^{n} \mathbf{P}[\,\mathcal{E}_i\,].$$

A Proof using Indicator Random Variables:

1. Let $\mathbf{1}_{\mathcal{E}_i}$ be the random variable that takes value 1 if $\mathcal{E}_i$ holds, 0 otherwise
2. $\mathbf{E}[\,\mathbf{1}_{\mathcal{E}_i}\,] = \mathbf{P}[\,\mathcal{E}_i\,]$ (**Check this**)
3. It is clear that $\mathbf{1}_{\bigcup_{i=1}^{n} \mathcal{E}_i} \leq \sum_{i=1}^{n} \mathbf{1}_{\mathcal{E}_i}$ (**Check this**)

Boole's Inequality (Union Bound)

Union Bound is one of the most basic probability inequalities, yet it is extremely useful and easy to apply!

Union Bound

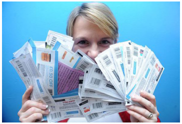
Let $\mathcal{E}_1, \ldots, \mathcal{E}_n$ be a collection of events in Σ. Then

$$\mathbf{P}\left[\bigcup_{i=1}^{n} \mathcal{E}_i\right] \leq \sum_{i=1}^{n} \mathbf{P}\left[\,\mathcal{E}_i\,\right].$$

A Proof using Indicator Random Variables:

1. Let $\mathbf{1}_{\mathcal{E}_i}$ be the random variable that takes value 1 if $\mathcal{E}_i$ holds, 0 otherwise
2. $\mathbf{E}\left[\,\mathbf{1}_{\mathcal{E}_i}\,\right] = \mathbf{P}\left[\,\mathcal{E}_i\,\right]$ (**Check this**)
3. It is clear that $\mathbf{1}_{\bigcup_{i=1}^{n} \mathcal{E}_i} \leq \sum_{i=1}^{n} \mathbf{1}_{\mathcal{E}_i}$ (**Check this**)
4. Taking expectation completes the proof.

Example: Coupon Collector

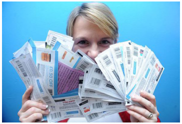


Source: `https://www.express.co.uk/life-style/life/567954/Discount-codes-money-saving-vouchers-coupons-mum`

Coupon Collector Problem

Suppose that there are n coupons to be collected from the cereal box. Every morning you open a new cereal box and get one coupon. We assume that each coupon appears with the same probability in the box.

Example: Coupon Collector



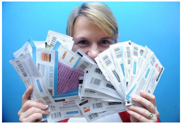
Source: `https://www.express.co.uk/life-style/life/567954/Discount-codes-money-saving-vouchers-coupons-mum`

This is a very important example in the design and analysis of randomised algorithms.

Coupon Collector Problem

Suppose that there are n coupons to be collected from the cereal box. Every morning you open a new cereal box and get one coupon. We assume that each coupon appears with the same probability in the box.

Example: Coupon Collector



Source: `https://www.express.co.uk/life-style/life/567954/Discount-codes-money-saving-vouchers-coupons-mum`

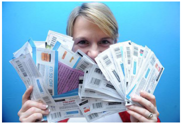
This is a very important example in the design and analysis of randomised algorithms.

Coupon Collector Problem

Suppose that there are n coupons to be collected from the cereal box. Every morning you open a new cereal box and get one coupon. We assume that each coupon appears with the same probability in the box.

Example Sequence for $n = 8$: $7, 6, 3, 3, 3, 2, 5, 4, 2, 4, 1, 4, 2, 1, 4, 3, 1, 4, 8$ ✓

Example: Coupon Collector



Source: `https://www.express.co.uk/life-style/life/567954/Discount-codes-money-saving-vouchers-coupons-mum`

This is a very important example in the design and analysis of randomised algorithms.

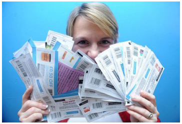
Coupon Collector Problem

Suppose that there are n coupons to be collected from the cereal box. Every morning you open a new cereal box and get one coupon. We assume that each coupon appears with the same probability in the box.

Example Sequence for $n = 8$: $7, 6, 3, 3, 3, 2, 5, 4, 2, 4, 1, 4, 2, 1, 4, 3, 1, 4, 8$ ✓

Exercise (Supervision)

Example: Coupon Collector



Source: `https://www.express.co.uk/life-style/life/567954/Discount-codes-money-saving-vouchers-coupons-mum`

This is a very important example in the design and analysis of randomised algorithms.

Coupon Collector Problem

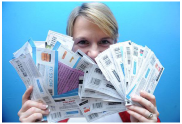
Suppose that there are n coupons to be collected from the cereal box. Every morning you open a new cereal box and get one coupon. We assume that each coupon appears with the same probability in the box.

Example Sequence for $n = 8$: $7, 6, 3, 3, 3, 2, 5, 4, 2, 4, 1, 4, 2, 1, 4, 3, 1, 4, 8$ ✓

Exercise (Supervision)

1. Prove it takes $n \sum_{k=1}^{n} \frac{1}{k} \approx n \log n$ expected boxes to collect all coupons

Example: Coupon Collector



Source: `https://www.express.co.uk/life-style/life/567954/Discount-codes-money-saving-vouchers-coupons-mum`

This is a very important example in the design and analysis of randomised algorithms.

Coupon Collector Problem

Suppose that there are n coupons to be collected from the cereal box. Every morning you open a new cereal box and get one coupon. We assume that each coupon appears with the same probability in the box.

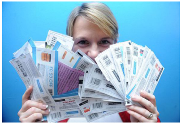
Example Sequence for $n = 8$: $7, 6, 3, 3, 3, 2, 5, 4, 2, 4, 1, 4, 2, 1, 4, 3, 1, 4, 8$ ✓

Exercise (Supervision)

In this course: $\log n = \ln n$

1. Prove it takes $n \sum_{k=1}^{n} \frac{1}{k} \approx n \log n$ expected boxes to collect all coupons

Example: Coupon Collector



Source: `https://www.express.co.uk/life-style/life/567954/Discount-codes-money-saving-vouchers-coupons-mum`

This is a very important example in the design and analysis of randomised algorithms.

Coupon Collector Problem

Suppose that there are n coupons to be collected from the cereal box. Every morning you open a new cereal box and get one coupon. We assume that each coupon appears with the same probability in the box.

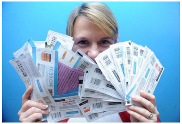
Example Sequence for $n = 8$: $7, 6, 3, 3, 3, 2, 5, 4, 2, 4, 1, 4, 2, 1, 4, 3, 1, 4, 8$ ✓

Exercise (Supervision)

In this course: $\log n = \ln n$

1. Prove it takes $n \sum_{k=1}^{n} \frac{1}{k} \approx n \log n$ expected boxes to collect all coupons
2. Use Union Bound to prove that the probability it takes more than $n \log n + cn$ boxes to collect all n coupons is $\leq e^{-c}$.

Example: Coupon Collector



Source: `https://www.express.co.uk/life-style/life/567954/Discount-codes-money-saving-vouchers-coupons-mum`

This is a very important example in the design and analysis of randomised algorithms.

Coupon Collector Problem

Suppose that there are n coupons to be collected from the cereal box. Every morning you open a new cereal box and get one coupon. We assume that each coupon appears with the same probability in the box.

Example Sequence for $n = 8$: $7, 6, 3, 3, 3, 2, 5, 4, 2, 4, 1, 4, 2, 1, 4, 3, 1, 4, 8$ ✓

In this course: $\log n = \ln n$

Exercise (Supervision)

1. Prove it takes $n \sum_{k=1}^{n} \frac{1}{k} \approx n \log n$ expected boxes to collect all coupons
2. Use Union Bound to prove that the probability it takes more than $n \log n + cn$ boxes to collect all n coupons is $\leq e^{-c}$.

   Hint: It is useful to remember that $1 - x \leq e^{-x}$ for all x