

CS 580: Algorithm Design and Analysis

Jeremiah Blocki
Purdue University
Spring 2019

Announcements: Homework 6 and Practice Final Exam Solutions Posted

Course Evaluation Survey: Live until Sunday (4/28/2019) at 11:59PM. Your feedback is valued!

Final Exam Logistics

- **Time:** Monday, April 29th at 8AM
- **Location:** PHYS 223 (adjacent building to FRNY)
- **Duration:** 2 hours
- **Content:** Cumulative, but more heavily focused on recent topics (e.g., PSPACE, Approximation Algorithms, Randomized Algorithms, Local Search etc...)
- No Electronics (calculator/phone/laptop/smartwatch etc...)
- **Index Cards:** We will allow you to bring **two** 3x5 index cards (double sided)
 - **Advice:** Don't expect to rely too heavily on your index card.
 - Effort spent preparing the index cards will likely be most beneficial.
- Practice Exam (Partial Solutions)

Practice Problem 1 (MAX SAT)

- MAX SAT (n variables, k clauses)
 - Each clause can have 1,2 or more literals
- **Part A:** Show that a random assignment satisfies *at least* $k/2$ clauses in expectation.

Practice Problem 1

- MAX SAT (n variables, k clauses)
 - Each clause can have 1,2 or more literals
- **Part B:** Suppose that we disallow directly contradictory clauses e.g., if $C = \{x\}$ is a clause then we cannot include the clause $C' = \{\bar{x}\}$. Modify the random assignment such that we satisfy at least $0.6k$ clauses in expectation.
- **Tempting Idea:** If (resp. $\{\bar{x}\}$) is a clause then set $x = 1$ (resp. $x = 0$).
- **Counter Example?**

Practice Problem 1

- MAX SAT (n variables, k clauses)
 - Each clause can have 1,2 or 3 literals
- **Part B:** Suppose that we disallow directly contradictory clauses e.g., if $C = \{x\}$ is a clause then we cannot include the clause $C' = \{\bar{x}\}$. Modify the random assignment such that we satisfy at least $0.6k$ clauses in expectation.
- **Tempting Idea:** If $\{x\}$ (resp. $\{\bar{x}\}$) is a clause then set $x = 1$ (resp. $x = 0$).
- **Counter Example?** $\{\bar{x}\}, \{\bar{y}\}, \{\bar{z}\}, \{x,y\}, \{x,z\}, \{y,z\}$

Practice Problem 1

- MAX SAT (n variables, k clauses)
 - Each clause can have 1,2 or more literals
- **Part B:** Suppose that we disallow directly contradictory clauses e.g., if $C = \{x\}$ is a clause then we cannot include the clause $C' = \{\bar{x}\}$. Modify the random assignment such that we satisfy at least $0.6k$ clauses in expectation.
- **More Refined Analysis:** Let y_i (denote the number of clauses with exactly i literals).
 - Observe that $k = \sum_i y_i$
 - **Linearity of Expectation:** We satisfy $\frac{1}{2}y_1 + \frac{3}{4}y_2 + \frac{7}{8}y_3 + \dots = k - \sum_i \frac{y_i}{2^i}$ clauses in expectation
 - If $y_1 \leq 0.6k$ then this is at least $\frac{1}{2}0.6k + \frac{3}{4}0.4k = 0.6k$ clauses in expectation.
 - What do we do otherwise?

Practice Problem 2

• Greedy Vertex Cover Algorithm

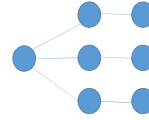
BuildVertexCover($G=(V,E)$)

- Initialize $S = \{\}$
- Find the node v with maximum degree in G
 - Recursively find a vertex cover S' for the graph $G - \{v\}$
 - $S' = \text{BuildVertexCover}(G - \{v\})$
 - Return $S = S' \cup \{v\}$

- **True or False:** Greedy Vertex Cover always returns the optimal vertex cover?

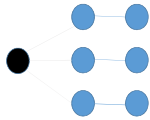
Practice Problem 2

- **True or False:** Greedy Vertex Cover always returns the optimal vertex cover?



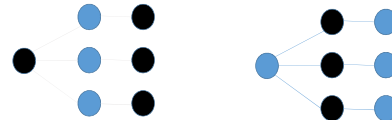
Practice Problem 2

- **True or False:** Greedy Vertex Cover always returns the optimal vertex cover?



Practice Problem 2

- **True or False:** Greedy Vertex Cover always returns the optimal vertex cover?



- **Answer:** False. Greedy returns vertex cover of size 4. OPT = 3.

Practice Problem 2'

• Greedy Triangle Cover Algorithm

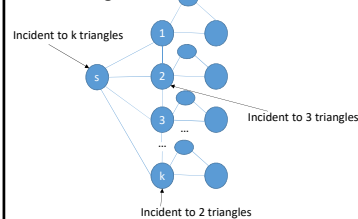
BuildTriangleCover($G=(V,E)$)

- Initialize $S = \{\}$
- Find the node v incident to maximum number of **triangles** in G
 - Recursively find a triangle cover S' for the graph $G - \{v\}$
 - $S' = \text{BuildTriangleCover}(G - \{v\})$
 - Return $S = S' \cup \{v\}$

- **True or False:** Greedy Triangle Cover always returns the optimal triangle cover?

Practice Problem 2

- **True or False:** Greedy Triangle Cover always returns the optimal triangle cover?



Practice Problem 2

- True or False:** Greedy Triangle Cover always returns the optimal triangle cover?

Practice Problem 2

- True or False:** Greedy Triangle Cover always returns the optimal triangle cover?

Practice Problem 3 (Approximate Median)

- Suppose we are presented with a very large set S of $n = |S|$ distinct real numbers and we want to approximate the median of these numbers by sampling. We say that a real number x is an ϵ -approximate median of S if at least $(\frac{1}{2} - \epsilon)$ numbers in S are less than x and at least $(\frac{1}{2} + \epsilon)$ numbers in S are greater than x .
- Suppose that we sub-sample $k = \frac{40}{\epsilon^2}$ points $y_1, \dots, y_k \in S$ from S (with replacement) and compute the median y_{med} of these points.
- Given that $0 < \epsilon < \frac{1}{2}$ show that the probability y_{med} is not an ϵ -approximate median of S is at most $\gamma = 0.0001$.
- Hint 1:** $(1 - \epsilon) \left(\frac{1}{2} + \epsilon\right) k \geq \frac{k}{2}$ for positive numbers $k > 0$ when $0 < \epsilon < \frac{1}{2}$
- Hint 2:** $e^{-10} \leq \frac{\gamma}{2} = 0.00005$ for positive numbers $k > 0$ when $0 < \epsilon < \frac{1}{2}$

Practice Problem 3 (Approximate Median)

- Suppose that we sub-sample $k = \frac{100}{\epsilon^2}$ points $y_1, \dots, y_k \in S$ from S (with replacement) and compute the median y_{med} of these points.
- What is the probability that y_{med} is an approximate median of S ?
- Definition:** Let x_{lowest} (resp. $x_{highest}$) denote the smallest (resp. largest) in S which is an ϵ -approximate median.
- Introduce Random Variable:** z_i which is 1 if and only if $y_i \geq x_{lowest}$
- Observation 1:** $E[z_i] = \Pr[z_i = 1] = \frac{1}{2} + \epsilon$
- Observation 2:**

$$\Pr[y_{med} \geq x_{lowest}] = \Pr\left[\sum_{i=1}^k z_i \geq \frac{k}{2}\right] = 1 - \Pr\left[\sum_{i=1}^k z_i < \frac{k}{2}\right]$$

Practice Problem 2 (Approximate Median)

Theorem. Suppose X_1, \dots, X_n are independent 0-1 random variables. Let $X = X_1 + \dots + X_n$. Then for any $\mu \leq E[X]$ and for any $0 < \delta < 1$, we have

$$\Pr[X < (1 - \delta)\mu] < e^{-\delta^2 \mu / 2}$$

Theorem. Suppose X_1, \dots, X_n are independent 0-1 random variables. Let $X = X_1 + \dots + X_n$. Then for any $\mu \geq E[X]$ and for any $\delta > 0$, we have

$$\Pr[X > (1 + \delta)\mu] < \frac{e^{-\delta^2 \mu}}{(1 + \delta)^{2\mu}}$$

Practice Problem 3 (Approximate Median)

- Suppose that we sub-sample $k = \frac{100}{\epsilon^2}$ points $y_1, \dots, y_k \in S$ from S (with replacement) and compute the median y_{med} of these points.
- Introduce Random Variable:** z_i which is 1 if and only if $y_i \geq x_{lowest}$
- Observation 1:** $E[z_i] = \Pr[z_i = 1] = \frac{1}{2} + \epsilon$
- Observation 2:** $\Pr[y_{med} \geq x_{lowest}] = 1 - \Pr\left[\sum_{i=1}^k z_i < \frac{k}{2}\right]$ **Hint 1:** $(1 - \epsilon) \left(\frac{1}{2} + \epsilon\right) k \geq \frac{k}{2}$ for positive numbers $k > 0$ when $0 < \epsilon < \frac{1}{2}$
- Chernoff Bound:** $\mu = \left(\frac{1}{2} + \epsilon\right) k \rightarrow \frac{k}{2} \leq \mu(1 - \epsilon)$

$$\Pr\left[\sum_{i=1}^k z_i < \frac{k}{2}\right] \leq \Pr\left[\sum_{i=1}^k z_i < \mu(1 - \epsilon)\right] \leq e^{-\frac{\epsilon^2 \mu}{2}} \leq e^{-\frac{\epsilon^2 k}{4}} \leq e^{-10} = \frac{\gamma}{2}$$

Practice Problem 3 (Approximate Median)

- Suppose that we sub-sample $k = \frac{100}{\varepsilon^2}$ points $y_1, \dots, y_k \in S$ from S (with replacement) and compute the median y_{med} of these points.
- **Introduce Random Variable:** z_i which is 1 if and only if $y_i > x_{lowest}$
- **Observation 1:** $E[z_i] = Pr[z_i = 1] = \frac{1}{2} + \varepsilon$
- **Observation 2:** $Pr[y_{med} \geq x_{lowest}] = 1 - Pr\left[\sum_{i=1}^k z_i < \frac{k}{2}\right]$

• **Conclusion:**

$$Pr[y_{med} \geq x_{lowest}] \geq 1 - e^{-10} = 1 - \frac{\gamma}{2}$$

Similar Argument:

$$Pr[y_{med} \leq x_{highest}] \geq 1 - e^{-10} = 1 - \frac{\gamma}{2}$$

Practice Problem 3 (Approximate Median)

- Suppose that we sub-sample $k = \frac{100}{\varepsilon^2}$ points $y_1, \dots, y_k \in S$ from S (with replacement) and compute the median y_{med} of these points.
- **Introduce Random Variable:** z_i which is 1 if and only if $y_i \leq x_{highest}$

• **Observation 1:** $E[z_i] = Pr[z_i = 1] = \frac{1}{2} + \varepsilon$

• **Observation 2:** $Pr[y_{med} \leq x_{highest}] = 1 - Pr\left[\sum_{i=1}^k z_i < \frac{k}{2}\right]$

• **Chernoff Bound:** $\mu = \left(\frac{1}{2} + \varepsilon\right)k \rightarrow \frac{k}{2} \leq \mu(1 - \varepsilon)$

$$Pr\left[\sum_{i=1}^k z_i < \frac{k}{2}\right] \leq Pr\left[\sum_{i=1}^k z_i < \mu(1 - \varepsilon)\right] \leq e^{-\frac{\varepsilon^2 \mu}{2}} \leq e^{-\frac{\varepsilon^2 k}{4}} = e^{-10}$$

Practice Problem 3 (Approximate Median)

• **Conclusion**

$$Pr[y_{med} \leq x_{highest}] = 1 - Pr\left[\sum_{i=1}^k z_i < \frac{k}{2}\right] \leq 1 - e^{-10}$$

$$Pr[y_{med} \geq x_{lowest}] \geq 1 - e^{-10}$$

$$Pr[y_{med} \text{ is not } \varepsilon\text{-approx}] \leq 2e^{-10} \leq 0.0001$$

Practice with ZPP and RP

- **Notation:** Given a randomized algorithm \mathcal{A} we write $y = \mathcal{A}(x; R)$ to denote the output of \mathcal{A} on input x fixing the random coins to be R .
- **ZPP:** A language X is in ZPP if there is an probabilistic polynomial time algorithm \mathcal{A} such that for all inputs x and all random strings R we have $Pr[\mathcal{A}(x; R) = 1 \mid x \in X] = 1$ and $Pr[\mathcal{A}(x; R) = 0 \mid x \notin X] = 1$.

• Show that $ZPP \subseteq NP$

- Certificate for $x \in X$: random string R such that $T(x; R) \leq p(|x|)$
 - $T(x; R)$ denotes running time of \mathcal{A} on input x with fixed random coins R
 - Probabilistic polynomial time $\rightarrow E[T(x)] \leq p(|x|) \rightarrow$ Short R exists

Random Variable: Running time of \mathcal{A} on input x

Practice with ZPP and RP

- Show that $ZPP \subseteq NP$
- Certificate for $x \in X$: random string R such that $T(x; R) \leq p(|x|)$
 - $T(x; R)$ denotes running time of \mathcal{A} on input x with fixed random coins R
 - Probabilistic polynomial time $\rightarrow E[T(x)] \leq p(|x|) \rightarrow$ Short R exists
- Certifier runs $\mathcal{A}(x; R)$ for $p(|x|)$
 - If $\mathcal{A}(x; R)$ returns 1 then output 1 (accept the proof that $x \in X$)
 - If $\mathcal{A}(x; R)$ returns 0 then output 0 (reject the proof that $x \in X$)
 - If $\mathcal{A}(x; R)$ does not halt then output 0 (reject the proof that $x \in X$)

Practice with ZPP and RP

- Show that $ZPP \subseteq coNP$?
- Certificate for $x \notin X$: random string R such that $T(x; R) \leq p(|x|)$
 - $T(x; R)$ denotes running time of \mathcal{A} on input x with fixed random coins R
 - Probabilistic polynomial time $\rightarrow E[T(x)] \leq p(|x|) \rightarrow$ Short R exists
- Certifier runs $\mathcal{A}(x; R)$ for $p(|x|)$
 - If $\mathcal{A}(x; R)$ returns 1 then output 0 (reject the proof that $x \notin X$)
 - If $\mathcal{A}(x; R)$ returns 0 then output 1 (accept the proof that $x \notin X$)
 - If $\mathcal{A}(x; R)$ does not halt then output 0 (reject the proof that $x \notin X$)

Practice with ZPP and RP

- Show that $RP \subseteq NP$?
- A language X is in RP if there is an **polynomial time** algorithm \mathcal{A} such that for all inputs x and all random strings R we have $\Pr[\mathcal{A}(x; R) = 1 \mid x \in X] \geq \frac{1}{2}$ and $\Pr[\mathcal{A}(x; R) = 0 \mid x \notin X] = 1$.
- Remark 1: For all x, R we have $T(x; R) \leq p(|x|)$ (**polynomial time**)
- Remark 2: One sided error. Allowed make mistakes when $x \in X$ (but not when $x \notin X$).
- Certificate: R such that $\mathcal{A}(x; R) = 1$
- **Claim:** If $x \notin X$ there is no valid certificate. Why?

More Complexity Theory

- Suppose that $NP = PSPACE$. Does it follow that $NP = coNP$?
- Answer: YES! PSPACE is closed under complementation.
- Though Question: What other complexity classes are closed under complementation?

ZPP and RP

- **Notation:** Given a randomized algorithm \mathcal{A} we write $y = \mathcal{A}(x; R)$ to denote the output of \mathcal{A} on input x fixing the random coins to be R .
- Running time is $T(x; R)$ with fixed random coins R
- **Remark:** Once x and R are fixed the output of $\mathcal{A}(x; R)$ is deterministic as is $T(\mathcal{A}; R)$.
- By Contrast, $\mathcal{A}(x)$ and $T(x)$ are both random variables.
- Probabilistic Polynomial Time: For all inputs x
- $E[T(x)] \leq p(|x|)$
- **ZPP:** A language X is in ZPP if there is a probabilistic polynomial time algorithm \mathcal{A} such that for all inputs x and all random strings R we have $\Pr[\mathcal{A}(x; R) = 1 \mid x \in X] = 1$ and $\Pr[\mathcal{A}(x; R) = 0 \mid x \notin X] = 1$

Feasible Subset Sum

- Feasible Subset Sum
 - **Input:** Set $A = \{a_1, \dots, a_n\}$ of positive integers and a positive integer $B > 0$.
 - Feasible Subset: $S \subset A$ is feasible if $\sum_{x \in S} x \leq B$.
 - **Goal:** Find feasible subset $S \subset A$ maximizing $\sum_{x \in S} x$
- **Example:** $A = \{8, 2, 4\}$ and $B = 11 \rightarrow S = \{8, 2\}$ is optimal.
- **Greedy Algorithm:**
 - Initialize: $S = \{\}$
 - For $i=1, \dots, n$
 - If $a_i + \sum_{x \in S} x \leq B$ then update $S = S \cup \{a_i\}$
- **Part 1:** Give an instance where greedy algorithm (above) is *not* a 2-approximation.

Feasible Subset Sum

- **Greedy Algorithm:**
 - Initialize: $S = \{\}$
 - For $i=1, \dots, n$
 - If $a_i + \sum_{x \in S} x \leq B$ then update $S = S \cup \{a_i\}$
- **Part 1:** Give an instance where greedy algorithm (above) is *not* a 2-approximation.
- **Example:** $A = \{2, 8\}$ and $B = 9$
 - Greedy Solution: $S = \{2\}$, but optimal is $S^* = \{8\}$ (four times better)

Feasible Subset Sum

- Feasible Subset Sum
 - **Input:** Set $A = \{a_1, \dots, a_n\}$ of positive integers and a positive integer $B > 0$.
 - Feasible Subset: $S \subset A$ is feasible if $\sum_{x \in S} x \leq B$.
 - **Goal:** Find feasible subset $S \subset A$ maximizing $\sum_{x \in S} x$
- **Example:** $A = \{8, 2, 4\}$ and $B = 11 \rightarrow S = \{8, 2\}$ is optimal.
- **Greedy Algorithm:**
 - Initialize: $S = \{\}$
 - For $i=1, \dots, n$
 - If $a_i + \sum_{x \in S} x \leq B$ then update $S = S \cup \{a_i\}$
- **Part 2:** Modify the above algorithm to yield a 2-approximation.

Feasible Subset Sum

- **Greedy Algorithm:**
 - Assumption we have eliminated any $a_i > B$
 - Sort $A = \{a_1, \dots, a_n\}$ so that $a_1 > a_2 > \dots$
 - Initialize: $S_0 = \{0\}$
 - For $i=1, \dots, n$
 - If $a_i + \sum_{x \in S_{i-1}} x \leq B$ then update $S_i = S_{i-1} \cup \{a_i\}$; otherwise $S_i = S_{i-1}$
- **Analysis:** We claim that either $S_n = A$ or $\sum_{x \in S_n} x \geq \frac{B}{2}$
- **Proof:** If $S_n \neq A$ we let j be first index such that $a_j + \sum_{x \in S_{j-1}} x > B$.
 - Case 1: $a_j < B$ we have $\sum_{x \in S_{j-1}} x > B - a_j > \frac{B}{2} \Rightarrow \sum_{x \in S_n} x > \frac{B}{2}$
 - Case 2: $a_j > \frac{B}{2}$ then we already added some earlier item $a_i > a_j \Rightarrow \sum_{x \in S_n} x > a_i > \frac{B}{2}$

Monotone Satisfiability Problem

- A *monotone* 3-SAT formula φ is specified by clauses C_1, \dots, C_k over variables x_1, \dots, x_n with the restriction that for each variable x_j its negation \bar{x}_j is never used in *any* clause.
- **Decision Problem 1:** Does φ have a satisfying assignment?
- **Question 1:** Is there a polynomial time algorithm to solve decision problem 1?

Monotone Satisfiability Problem

- A *monotone* 3-SAT formula φ is specified by clauses C_1, \dots, C_k over variables x_1, \dots, x_n with the restriction that for each variable x_j its negation \bar{x}_j is never used in *any* clause.
- **Decision Problem 2:** Does φ have a satisfying assignment in which *at most* k -variables are set to 1?
- **Question 2:** Show that this second decision problem is NP-Complete.
- Step 1?

Monotone Satisfiability Problem

- A *monotone* 3-SAT formula φ is specified by clauses C_1, \dots, C_k over variables x_1, \dots, x_n with the restriction that for each variable x_j its negation \bar{x}_j is never used in *any* clause.
- **Decision Problem 2:** Does φ have a satisfying assignment in which *at most* k -variables are set to 1?
- **Question 2:** Show that this second decision problem is NP-Complete.
- **Step 1:** Show that decision problem 2 is in NP.
- **Witness:** satisfying assignment with at most k -variables set to 1.

Monotone Satisfiability Problem

- A *monotone* 3-SAT formula φ is specified by clauses C_1, \dots, C_k over variables x_1, \dots, x_n with the restriction that for each variable x_j its negation \bar{x}_j is never used in *any* clause.
- **Decision Problem 2:** Does φ have a satisfying assignment in which *at most* k -variables are set to 1?
- **Question 2:** Show that this second decision problem is NP-Complete.
- **Step 2:** Reduction from known NP-Complete Problem
- **Hint:** Try vertex cover

Monotone Satisfiability Problem

- A *monotone* 3-SAT formula φ is specified by clauses C_1, \dots, C_k over variables x_1, \dots, x_n with the restriction that for each variable x_j its negation \bar{x}_j is never used in *any* clause.
- **Decision Problem 2:** Does φ have a satisfying assignment in which *at most* k -variables are set to 1?
- **Question 2:** Show that this second decision problem is NP-Complete.
- **Step 2:** Reduction from known NP-Complete Problem
- **Vertex Cover Instance:** (G, k)
- **Monotone 3-SAT Formula:** φ_G

Monotone Satisfiability Problem

- A *monotone* 3-SAT formula φ is specified by clauses C_1, \dots, C_k over variables x_1, \dots, x_n with the restriction that for each variable x_j its negation \bar{x}_j is never used in *any* clause.
- **Decision Problem 2:** Does φ have a satisfying assignment in which at most k -variables are set to 1?
- **Question 2:** Show that this second decision problem is NP-Complete.
- **Step 2:** Reduction from known NP-Complete Problem
- **Vertex Cover Instance:** (G, k)
- **Monotone 3-SAT Formula:** φ_G
 - Add Variable x_v for each node and clause $C_e = \{x_u, x_v\}$ for each edge $e=\{u,v\}$

Monotone Satisfiability Problem

- **Step 2:** Reduction from known NP-Complete Problem
- **Vertex Cover Instance:** (G, k)
- **Monotone 3-SAT Instance $f(G, k): \varphi_G$ and k**
 - Build φ_G : Add Variable x_v for each node and clause $C_e = \{x_u, x_v\}$ for each edge $e=\{u,v\}$
- **Claim 1: If G has a vertex cover of size k then φ_G has a satisfying assignment in which at most k variables are true.**
 - **Proof:** Let S be vertex cover and then set $x_u = 1$ for each node $u \in S$. For each clause C_e we either have $u \in S$ or $v \in S$ and hence the clause is satisfied.

Monotone Satisfiability Problem

- **Step 2:** Reduction from known NP-Complete Problem
- **Vertex Cover Instance:** (G, k)
- **Monotone 3-SAT Instance $f(G, k): \varphi_G$ and k**
 - Build φ_G : Add Variable x_v for each node and clause $C_e = \{x_u, x_v\}$ for each edge $e=\{u,v\}$
- **Claim 2: If φ_G has a satisfying assignment in which at most k variables are true then G has a vertex cover of size k .**
 - **Proof:** Given satisfying assignment we define a vertex cover S in which we add $u \in S$ if and only if $x_u = 1$. For each clause C_e we either have $x_u = 1$ or $x_v = 1$ and hence each edge e is covered by S .

More Reductions

- Suppose that we have a polynomial time Karp reduction from decision problem X to decision problem Y i.e. $X \leq_p Y$.
- Which of the following claims are **necessarily** true?
 - A. If Y is in PSPACE then X is in PSPACE
 - B. If Y is PSPACE-Complete then X is in PSPACE
 - C. If Y is NP-Complete then X is NP-Complete
 - D. If Y is NP-Complete then X is NP-Hard
 - E. If X is NP-Complete then Y is NP-Complete
 - F. If Y is in P then X is in P
 - G. If Y is in ZPP then X is in ZPP.