Problem Set 6

Graded

Student

Anton Melnychuk

Total Points

40 / 41 pts

Question 1

Your Information

1 / 1 pt

- ✔ **+ 0.2 pts** Name
- ✔ **+ 0.2 pts** SID
- ✔ **+ 0.2 pts** Collaborators and resources
- ✔ **+ 0.2 pts** Academic integrity policy
- ✔ **+ 0.2 pts** Hours spent

Question 2

Discrete Quadratic Minimization

19 / 20 pts

✔ **+ 5 pts** DQM is in NP

Correct reduction

✔ **+ 2 pts** Correctly identified "summation" term $\sum_i x_i$ counting vertices

✔ **+ 2 pts** Correctly identified "uncovered" term $C \cdot \sum_{u,v} q_{u,v}$ where $C > k$

+ 1 pt Claimed that the reduction can be completed in polynomial time

---

(Completeness) If Vertex-Cover instance has a *Yes* answer then DQM has a *Yes* answer

✔ **+ 2 pts** Correct mapping from vertex cover to $(x_1, \ldots, x_n)$

✔ **+ 2 pts** If vertex cover is valid then the "uncovered" term $C \cdot \sum_{u,v} q_{u,v} = 0$

✔ **+ 1 pt** If vertex cover has size at most k, "summation" term $\sum_i x_i \leq k$

(Soundness) If DQM instances has a *Yes* answer then Vertex-Cover has a *Yes* answer

✔ **+ 2 pts** Correct mapping from $(x_1, \ldots, x_n)$ to vertex cover V'

✔ **+ 2 pts** If $x^\star$ is the minimizer of the polynomial Q and $Q(x^\star) \leq T$, then $C \cdot \sum_{u,v} q_{u,v} = 0$, and, hence, V' is a valid vertex cover

✔ **+ 1 pt** If $min_x Q(x) \leq T$, then V' is a vertex cover of size at most k

---

+ 0 pts Incorrect

Question 3

Scheduling

20 / 20 pts

✔ **+ 5 pts** Weekend-Plan is in NP

– 1 pt Minor error or lack of rigor in NP inclusion justification

– 2 pts Moderate error or lack of rigor in NP inclusion justification

– 3 pts Significant error or lack of rigor in NP inclusion justification

---

Correct reduction

✔ **+ 1 pt** Correctly identified that the number of clauses is m (the number of friends)

✔ **+ 1 pt** Correctly identified that the number of variables is n (the number of weekends)

✔ **+ 4 pts** Correct construction of the matrix S

✔ **+ 1 pt** Claimed that the reduction can be completed in polynomial time

---

(Completeness) If SAT instance has a *Yes* answer then Weekend-Plan has a *Yes* answer

✔ **+ 2 pts** Correct mapping from a satisfying solution x for SAT to a valid weekend plan p

✔ **+ 1 pt** Prove the case where $S_{ij} = Saturday$

✔ **+ 1 pt** Prove the case where $S_{ij} = Sunday$

– 1 pt Minor error or lack of rigor in explanation

– 2 pts Moderate error or lack of rigor in explanation

– 3 pts Significant error or lack of rigor in explanation

(Soundness) If Weekend-Plan instances has a *Yes* answer then SAT has a *Yes* answer

✔ **+ 2 pts** Correct mapping from a valid weekend plan p to a satisfying solution x for SAT

✔ **+ 1 pt** Prove the case where $x_i \in C_j$ (for each clause C_j)

✔ **+ 1 pt** Prove the case where $\neg x_i \in C_j$ (for each clause C_j)

– 1 pt Minor error or lack of rigor in explanation

– 2 pts Moderate error or lack of rigor in explanation

– 3 pts Significant error or lack of rigor in explanation

– 1 pt Implies but does not directly explain how to assign values to variables

Question assigned to the following page: 1

CPSC/ECON 365: Algorithms Yale University

Solution to Problem Set 6

Student Name: November 27, 2023 Due: Monday, December 4, 2023 at 2:30 pm ET

0 Your Information

(a) Your name.

Anton Melnychuk

(b) Your SID.

24787491

(c) A list your collaborators and any outside resources you consulted for this problem set.

TAs: Christian, Dylan, Ryan

(d) Copy: "I have followed the academic integrity and collaboration policy as written above."

I have followed the academic integrity and collaboration policy as written above.

(e) How many hours did you spend in this problem set?

18 hours

No questions assigned to the following page.

1 Discrete Quadratic Minimization

Recall from your math class that we can minimize a quadratic function over real numbers easily. On the other hand, minimizing a quadratic function over a discrete set turns out to be difficult.

A **quadratic polynomial** Q is a function of $x_1, \ldots, x_n \in \{0, 1\}$ of the form:

$$Q(x_1, \ldots, x_n) = a + \sum_{i=1}^{n} b_i x_i + \sum_{i,j=1}^{n} c_{i,j} x_i x_j$$

for some integer coefficients $a, b_i, c_{i,j} \in \mathbb{Z}$, for $1 \leq i, j \leq n$. In the **Discrete Quadratic Minimization (DQM)** problem, you are asked to determine whether there is a choice of input that makes the quadratic polynomial small.

Below, the inputs are integer coefficients $a \in \mathbb{Z}, b = (b_1, \ldots, b_n) \in \mathbb{Z}^n, c = (c_{i,j})_{i,j=1}^{n} \in \mathbb{Z}^{n \times n}$ that determine a quadratic polynomial Q, and a threshold $T \in \mathbb{Z}$.

> **DQM**(a, b, c, T)**:** Return `Yes` if there exist $x_1, \ldots, x_n \in \{0, 1\}$ such that $Q(x_1, \ldots, x_n) \leq T$; return `No` otherwise.

Problem: Prove that **DQM** is NP-Complete.

(*Hint:* Try a reduction from **Vertex-Cover**.)

Question assigned to the following page: 2

Solution. Consider two parts of the proof.

1. DQM $\in NP$. The certifier would be checking if a given certificate inputs $x_1, \dots x_n$ with a given values of $a, b_i, c_{i,j} \in \mathcal{Z}$, for $1 \leq i, j \leq n$ and T satisfy the quadratic polynomial function $Q(x_1, \dots, x_n)$. This involves summation $(a+)$ constant time, $(\sum_{i=1}^{n})$ loop size of n with a multiplication in each iteration $b_i x_i$, which overall requires $O(n)$ time. Finally, $(\sum_{i,j=1}^{n} c_{i,j} x_i x_j)$ involves all possible combination (two nested loops both size of n), which is overall $O(n^2)$ time. The certifier also would take $O(n)$ time to ensure all inputs are valid and either $0, 1$. Therefore, overall complexity is $O(n^2 + 2n + 1) = O(n)$, which is a polynomial function.

2. DQM is $NP - Hard$. By reduction from the Vertex Cover problem:

VC($G(V, E), k$): Return **Yes** if there exists a set of vertices `U` that minimizes $\sum_{v \in V} x_v \leq k$ subject to $x_u + x_v \geq 1$ for every $\{u, v\} \in E$ and $x_v \in \{0, 1\}$ for all $v \in V$; **No** otherwise.

Consider an observation that $x_i + x_j - x_i x_j = 1$ holds $\forall i, j.\ (i, j) \in E$ using the Vertex Cover problem. Without lost of generality, enumerate each vertex. Now, given the graph G, for each vertex $v \in V$, introduce a binary variable x_v by:

$$\begin{aligned}[1^2 - (x_i + x_j) + x_i x_j)] &= 0 \\ (1 - x_i)(1 - x_j) &= 0\end{aligned}$$

Similarly, for all edges in G:

$$\sum_{(i,j) \in E} (1 - x_i)(1 - x_j) = 0\ ,\ \forall (i, j) \in E$$

Let's prove that $\sum_{(i,j) \in E} (1 - x_i)(1 - x_j) = 0 \iff \forall (i, j) \in E.\ x_i \vee x_j = 1$.

1. $\implies \forall (i, j) \in E,\ x_j \vee x_i = 1$. Otherwise, if $x_i \wedge x_j = 0$, then $\exists z = (1 - x_i)(1 - x_j) = 1$, so sum would we at least 1 (by the def of the sum), which contradicts the assumed result 0.

2. $\impliedby$ Since $x_i \vee x_j = 1$ for every term in the sum, the product $(1 - x_i)(1 - x_j)$ is 0 for every term where $(i, j) \in E$. Therefore, the sum $\sum_{(i,j) \in E} (1 - x_i)(1 - x_j)$ is the sum of 0s.

Since in the proof we expect the DQM to hold, to prevent the graph after the Vertex Cover have edges that does not exist or satisfy $x_i \vee x_j = 1$ constraint, let's set the input value of the

$$c_{i,j} = \begin{cases} (k+1) & \text{if } (i, j) \in E, \\ 0 & \text{otherwise.} \end{cases}$$

since we expect to receive $\sum_{i=1}^{n} x_i \leq k$ and if $\sum_{(i,j) \in E} (1 - x_i)(1 - x_j) > 0$, then since $(k+1) > T = k$, we will guarantee that if DQM holds, all edges are satisfiable and $\sum_{(i,j) \in E} (1 - x_i)(1 - x_j) = 0$ too.

Question assigned to the following page: 2

To receive the inputs b_i and a, consider the following transformation under the $c_{i,j}$ constraint:

$$\sum_{i=1}^{n} x_i + \sum_{(i,j)\in E}^{n} (k+1)[(1-x_i)(1-x_j)] \leq k$$

$$\sum_{i=1}^{n} x_i + \sum_{(i,j)\in E}^{n} (k+1)[1-x_i-x_j+x_ix_j] \leq k$$

Here, we want to leave the x_ix_j values only, then since we are working under the constraint of edges, we are able to mark them in the next way:

$$2|E|(k+1) + \sum_{i=1}^{n} x_i - \sum_{i=1}^{n} [deg(i)](k+1)x_i - \sum_{i=1}^{n} [deg(i)](k+1)x_i + \sum_{(i,j)\in E}^{n} (k+1)[x_ix_j] \leq k$$

(here in $2|E|(k+1)$, $2|E|$ by Hand-Shake Lemma)

$$\text{Algorithm X (notation)} \ = |E|(k+1) + \sum_{i=1}^{n} (1-2[deg(i)](k+1))x_i + \sum_{(i,j)\in E}^{n} (k+1)[x_ix_j] \leq k$$

Now, consider the $(a, b_i, c_{i,j}, T)$ instance of the DQM problem.

$$a = |E|(k+1)$$

$$b_i = 1 - 2[deg(i)](k+1)$$

$$c_{i,j} = \begin{cases} (k+1) & \text{if } (i,j) \in E, \\ 0 & \text{otherwise.} \end{cases}$$

$$T = k$$

Given the reduction algorithm, let's prove Algorithm X holds $\iff$ Vertex-Cover returns **Yes**.

1. $\implies$ If the provided algorithm holds, then it may happen only if Vertex-Cover is satisfied (by the construction of the problem). Consider the opposite, let (simplified; without inputs) $\exists (i,j) \in E$. s.t. $x_i \wedge x_j = 0$, then since $c_{i,j} = (k+1)$, if there exists positive output from the $(\sum_{(i,j)\in E}^{n} (1-x_i)(1-x_j))$, then $(k+1) * \cdot > k$ or generally $\sum_{i=1}^{n} x_i + \sum_{(i,j)\in E}^{n} (k+1)[(1-x_i)(1-x_j)] > k$, which leads to the contradiction, since we assumed that the algorithm holds and $\sum_{i=1}^{n} x_i + \sum_{(i,j)\in E}^{n} (k+1)[(1-x_i)(1-x_j)] \leq k$ is true.

Question assigned to the following page: 2

2. $\Longleftarrow$ Since, the Vertex-Cover holds, then $\forall(i,j) \in E.\ x_i \vee x_j = 1 \implies$

$$\sum_{(i,j)\in E}^{n} (k+1)[(1-x_i)(1-x_j)] = 0$$

and generally,

$$\sum_{i=1}^{n} x_i + \sum_{(i,j)\in E}^{n} (k+1)[(1-x_i)(1-x_j)] \leq k$$

since we know by the assumption that

$$\sum_{i=1}^{n} x_i \leq k$$

then $k \leq k$, which is true and the whole Algorithm X holds with the inputs produced by expanding the input-c parentheses, so they are the same:

$$2|E|(k+1) + \sum_{i=1}^{n} (1 - 2[deg(i)](k+1))x_i + \sum_{(i,j)\in E}^{n} (k+1)[x_i x_j] \leq k$$

■

No questions assigned to the following page.

2 Scheduling

In the Spring 2024 semester, there will be n weekends and, on each weekend, there will be one party on Saturday and one party on Sunday. Since you also have to study, you have decided to attend *exactly* one party each weekend. You have m friends $f_1, f_2, \ldots, f_m$. All of your friends have planned which parties they will attend (they are not bound to attend exactly one party each weekend). The schedule of your friends is captured by a matrix $S \in \{\texttt{Saturday}, \texttt{Sunday}, \texttt{Both}, \texttt{None}\}^{m \times n}$: for each friend f_j and each weekend w_i, the entry S_{ij} is:

$$S_{ij} = \begin{cases} \texttt{Saturday} & \text{if } f_j \text{ will } \textit{only} \text{ attend the Saturday party on weekend } w_i, \\ \texttt{Sunday} & \text{if } f_j \text{ will } \textit{only} \text{ attend the Sunday party on weekend } w_i, \\ \texttt{Both} & \text{if } f_j \text{ will attend both parties on weekend } w_i, \\ \texttt{None} & \text{if } f_j \text{ will attend neither party on weekend } w_i. \end{cases}$$

You want to ensure that you attend at least one party with each of your friends in the Spring 2024 semester.

Problem: Prove that, given S, it is NP-complete to determine whether there is a schedule in which (1) you attend exactly one party each weekend and (2) also attend at least one party with each of your friends during the semester.

(*Hint:* Try a reduction from **SAT**.)

Question assigned to the following page: 3

Solution. Consider two parts of the solution.

1. S is N$\mathcal{P}$. The certifier needs to satisfies two conditions: a. I attend exactly one party each weekend. b. I attend at least one party with each of your friends during the semester. Therefore, given a certificate, the certifier would apply the given inputs (set of the weeks $p_1 \dots p_n$) if it's binary choice: either go on Saturday or Sunday. To check the second part, the certifier would run through all the weeks and combine all the unique friends I have visited. Finally, check if the set of sets I have visited is equal to the input set of all friends. Checking condition 1 could be done along the iteration in $O(1)$, since if statement only, the second condiction would involve $O(nm)$ time if all friends m visit all parties I visit each weekend n. Therefore the overall time is $O(1+nm) = O(nm)$.

2. S is N$\mathcal{P}$-Hard. To reduce scheduling to the SAT, consider the following algorithm. Since, we aim to visit each friend f_j s.t. $1 \leq j \leq m$ at least once, let's map the clause to each friend that has certain preferences regarding when he is going to a party. Then, let the literal x_k s.t $1 \leq k \leq n$ be the week my friend is going to a party. Given an expression, let's denote x_k mean my friend is going to a party on Saturday (week k); $x\bar{}_k$ mean my friend is going to a party on Sunday (week k), since it's a binary choice. Then, to build a schedule from SAT, consider the following algorithm. For each of the parentheses choose its represented friend column; for each literal i, choose its represented weekend row and place the value in the following way:

$$S_{i,j} = \begin{cases} \texttt{None} & \text{if some } x_i \notin \mathcal{C}_j \wedge \bar{x}_i \notin \mathcal{C}_j, \\ \texttt{Both} & \text{if some } \bar{x}_i \in \mathcal{C}_j \wedge x_i \in \mathcal{C}_j, \\ \texttt{SUN} & \text{if some } \bar{x}_i \in \mathcal{C}_j, \\ \texttt{SAT} & \text{if some } x_i \in \mathcal{C}_j. \end{cases}$$

Proof. To prove it, let's show that my Scheduling instance $\iff$ Solution for the SAT problem.

1. $\implies$ For each weekend w_i in my schedule returned, let's assign truth values to the variables based on the parties I attend on that weekend (Want: SAT works). If I attend the Saturday party, set the corresponding variable x_i to true, since all friends' clause that also go to that party on ith week Saturday will return true. If I attend the Sunday party, set $\bar{x}_i$ to true, since all friends' clause that also go to that party on ith week Sunday will return true. Finally, if both values in the Scheduling return True for some friend, we are guaranteed to have True output regardless the choice I will make for SAT. Therefore it satisfies both (1) and (2) conditions. Since the schedule assumes that I attend exactly one party each weekend and at least one party with each of my friends, the constructed assignment to SAT also returns True. To support it, consider, by contradiction that it does not, then some clause in SAT and its all literals evaluate to False. It means there cannot be x_i and $\bar{x}_i$ in one clause (not Both),

Question assigned to the following page: 3

and by the construction, assumption corresponds to attending both parties on the i-th week, which contradicts the assumption of attending only one party each weekend.

2. $\Longleftarrow$ Construct the schedule P (notation) based on the satisfying scheduling provided, where $P = \{p_1, p_2..., p_n\}$ for each week p_i. To do that, first, use SAT to construct the $S_{i,j}$, then:

$$p_i = \begin{cases} \texttt{SAT} & \text{if solution to SAT } x_i = 1 \text{ (True)} \\ \texttt{SUN} & \text{if solution to SAT } x_i = 0 \text{ (False)} \end{cases}$$

Thus, if the SAT expression is True, we ensure that for each friend f_j associated with a clause $\mathcal{C}_j$ in the Scheduling problem I 1. attend at least one party with them and 2. attend exactly one party each weekend. ■