Introduction to the Theory of Computation 2023 — Final

Solutions

Problem 1 (25 pts). Assume f and g are functions $f, g : \mathbb{N} \to \mathbb{R}^+$. Prove or disprove the following statements by using the following definitions.

Definition 1. We say $f(n) = O(g(n))$ if there exists $c > 0$ and $n_0 \in \mathbb{N}$ such that for every integer $n \geq n_0$,

$$f(n) \leq cg(n).$$

Definition 2. We say $f(n) = o(g(n))$ if for each $c > 0$, there exists $n_0 \in \mathbb{N}$ such that for every integer $n \geq n_0$,

$$f(n) \leq cg(n).$$

(a) (10 pts) Let $f(n) = n!$ and $g(n) = 2^n$. $f(n) = O(g(n))$

(b) (10 pts) Let $f(n) = n!$ and $g(n) = n^n$. $f(n) = o(g(n))$

(c) (5 pts) Let $f(n) = \log(n!)$ and $g(n) = n \log n$. $f(n) = O(g(n))$

For proving the statements, you need to show the existence of n_0 for one c or all c's, depending on the definition of big-O or small-o. For disproving the statements, you must prove the opposite of the definition by also showing details.

Solution.

(a) The opposite of Definition 1 is

$$\text{for all } c > 0 \text{ and } n_0 \in \mathbb{N}, \text{ there exists an } n \geq n_0 \text{ such that } f(n) > cg(n). \qquad (\Delta)$$

For any $0 < c < 3$ and $n_0 \in \mathbb{N}$, there exists $n \geq \max\{n_0, 5\}$ such that

$$\frac{n!}{2^n} = \frac{n(n-1)\cdots(6)}{2 \cdot 2 \cdots 2} \cdot \frac{5 \cdot 4 \cdot 3 \cdot 2 \cdot 1}{2 \cdot 2 \cdot 2 \cdot 2 \cdot 2} > 1 \cdot 3 > c.$$

For any $c \geq 3$ and $n_0 \in \mathbb{N}$, there exists $n \geq \max\{\lceil 2c \rceil, n_0\}$ such that

$$\frac{n!}{2^n} = \frac{n}{2} \cdot \frac{(n-1)(n-2)\cdots 1}{2 \cdot 2 \cdots 2} > c \cdot 1 = c.$$

Combining both cases, we prove the opposite of Definition 1. Thus, the statement is disproved.
Common mistakes:

- Some wrongly think that the opposite of Definition 1 is

$$\text{for all } c > 0, \text{ there exists } n_0 \in \mathbb{N} \text{ such that } \forall n \geq n_0,\ f(n) > cg(n) \qquad (*)$$

  Interestingly, $(*)$ implies (Δ) by the following proof:
  Let us rename n_0 in $(*)$ to $\overline{n_0}$. Then in proving (Δ), we choose $n = \max\{n_0, \overline{n_0}\}$. Next we discuss the proof of $(*)$. For any $0 < c < 3$, choose $n_0 = 5$ and we have

$$\frac{n!}{2^n} = \frac{n(n-1)\cdots(6)}{2\cdot 2\cdots 2} \cdot \frac{5\cdot 4\cdot 3\cdot 2\cdot 1}{2\cdot 2\cdot 2\cdot 2\cdot 2} > 1\cdot 3 > c, \text{ for all } n \geq \overline{n_0}.$$

  For $c \geq 3$, choose $n_0 = \lceil 2c \rceil$ and we have

$$\frac{n!}{2^n} = \frac{n}{2} \cdot \frac{(n-1)(n-2)\cdots 1}{2\cdot 2\cdots 2} > c\cdot 1 = c, \text{ for all } n \geq \overline{n_0}.$$

  You are deducted some points if you give the wrong oppisite statement.
- Your proof must be clearly written. Steps of your proof must be logical.

(b) For any $c \geq 1$, we simply choose $n_0 = 1$ so that for $n \geq n_0$,

$$\frac{n!}{n^n} \leq 1.$$

For any $0 < c < 1$, we choose $n_0 = \lceil \frac{1}{c} \rceil$ so that for $n \geq n_0$,

$$\frac{n!}{n^n} = \frac{n(n-1)\cdots 2}{n\cdot n\cdots n} \cdot \frac{1}{n} \leq 1\cdot c.$$

Therefore, $n! \leq c\cdot n^n$. Combining both cases, we have $f(n) = o(g(n))$.

Comments: Some try to argue that

$$\lim_{n\to\infty} \frac{n!}{n^n} = 0.$$

But the definition of limit is Definition 2 and we specifically ask you to discuss c, n_0, n etc.

(c) Take $c = 1$ and $n_0 = 1$. We have for $n \geq n_0$,

$$f(n) = \log(n!) = \sum_{i=1}^{n} \log i \leq 1\cdot(n\log n) = c\cdot g(n),$$

showing that $f(n) = O(g(n))$.

Problem 2 (25 pts). In this problem, you have to design a two-tape TM for multiplying two non-zero polynomials with binary coefficients. For example,

$$(1+x)\times(1+x+x^2) = 1+x^3, \qquad (1)$$

where

$$1+1=0$$

in the binary number system. The format of input strings is defined by

$$a_0a_1\cdots a_n\#b_0b_1\cdots b_m,$$

where

$$a_i, b_j \in \{0, 1\}\ \forall i, j,$$

and

$$a_n = b_m = 1.$$

Your TM should locate the result in the 2nd tape. That is, if the input string is

$$11\#111,$$

which is the example (1), the 2nd tape of the TM should be

$$1 \times (1 + x + x^2) + x \times (1 + x + x^2) \Rightarrow 111 \sqcup +0111 = 1001. \tag{2}$$

Our idea is to calculate

$$a_i x^i (b_0 + b_1 x + \cdots + b_m x^m) \tag{3}$$

and add the corresponding coefficients of the result into the 2nd tape, for $i = 0, \ldots, n$. For example, in the example (1), after handling a_0, the resulting configuration should be

$$\begin{bmatrix} 1(\text{processed}) & 1 & \# & 1 & 1 & 1 \\ 1 & & 1 & 1 & \sqcup & \sqcup & \sqcup \end{bmatrix}$$

After writing the results of (3) to the 2nd tape, we should go to the next position for a_{i+1}. Specifically, in the 1st tape, we must know a_{i+1}, while in the 2nd tape, we must know the starting position for addition (i.e., the position of the coefficient for x^{i+1}). For example, after processing a_0, we consider $a_1 x$ and want to have the following configuration (here we use a dot to indicate the head position).

$$\begin{bmatrix} 1(\text{processed}) & \dot{1} & \# & 1 & 1 & 1 \\ 1 & & \dot{1} & 1 & \sqcup & \sqcup & \sqcup \end{bmatrix}$$

But the question is how to move the two heads to the desired positions. Eariler in finishing (3) for $a_i x^i$, the head of the 1st tape must be at the segment of $b_0 \ldots b_m$. To find a_{i+1}, our strategy is to mark the first position (i.e., a_0) by F and all other processed $a_1, \ldots, a_i$ by P. Then by moving both heads to the beginning of the tapes and then moving both heads right to pass over the first F and all P's in the first tape, our two heads are at the position of a_{i+1} (or say position of x^{i+1} in the second tape). Note that while we can rely on the last P to detect a_i and a_{i+1} in the first tape, we have difficulties to find the same position in the second tape. Thus we align the two heads by moving them to the beginning of the tapes. Here we give an illustration: After processing a_0 in (2), the configuration becomes

$$\begin{bmatrix} F & 1 & \# & 1 & 1 & 1 & \dot{\sqcup} \\ 1 & 1 & 1 & \dot{\sqcup} & \sqcup & \sqcup \end{bmatrix}$$

Then we move both heads to the beginning (by detecting F):

$$\begin{bmatrix} \dot{F} & 1 & \# & 1 & 1 & 1 & \sqcup \\ \dot{1} & 1 & 1 & \sqcup & \sqcup & \sqcup \end{bmatrix}$$

and we move right to pass the first F and all P's in the 1st tape (and every 0/1 in the 2nd):

$$\begin{bmatrix} F & \dot{1} & \# & 1 & 1 & 1 & \sqcup \\ 1 & \dot{1} & 1 & \sqcup & \sqcup & \sqcup \end{bmatrix}$$

This leads to the desired position.

Let us use the following algorithm to achieve our idea. We set

$$i = 0,$$

and execute the following steps.

Step 1: Read the current coefficient a_i in the 1st tape. If $i = 0$, we modify a_0 to F for marking the first position. Else ($i \geq 1$), we modify a_i to P. At this moment, the head of the 2nd tape should be at the position of x^i (see Step 5; also when $i = 0$, the 2nd head is rightly in the beginning of the tape.)

If $a_i = 1$, do Step 2 and Step 3. Otherwise ($a_i = 0$), go to Step 4.

Step 2: We move the 1st header to point at b_0, and the 2nd header should stay in the corresponding position of a_i.

Step 3: Calculate

$$a_i x^i (b_0 + b_1 x + \cdots + b_m x^m)$$

and add results to the corresponding positions in the 2nd tape.

Step 4: To process the next coefficient a_{i+1}, move both headers to the first position of each tape.

Step 5: Then, both headers keep moving right if the header of the 1st tape reads

$$F \text{ or } P.$$

After that, the 1st header should point at a_{i+1}, and the 2nd header points at the position corresponding to the coefficient of x^{i+1}.

Step 6: If the 1st header points at #, the multiplication is done and we should go to q_a. Else, update the index

$$i \leftarrow i + 1$$

and go to Step 1.

(a) (15 pts) Please design a two-tape Turing machine with

- ≤ 7 states (including q_a and q_r). We let $Q = \{q_0, q_1, q_2, q_3, q_4, q_a, q_r\}$ and q_0 be the start state
- $\Sigma = \{1, 0, \#\}$
- $\Gamma = \{1, 0, \#, F, P\}$

Note that S is allowed in a multi-tape Turing machine. **Links to q_r do not need to be drawn.** To simplify your diagram, you can use the format such as

$$\begin{cases} \{0, 1\} \to S \\ \{0, \#\} \to R \end{cases}$$

to represent the transition rules

$$\begin{cases} 0 \to S \\ 0 \to R \end{cases}, \begin{cases} 1 \to S \\ 0 \to R \end{cases}, \begin{cases} 0 \to S \\ \# \to R \end{cases}, \begin{cases} 1 \to S \\ \# \to R \end{cases}.$$

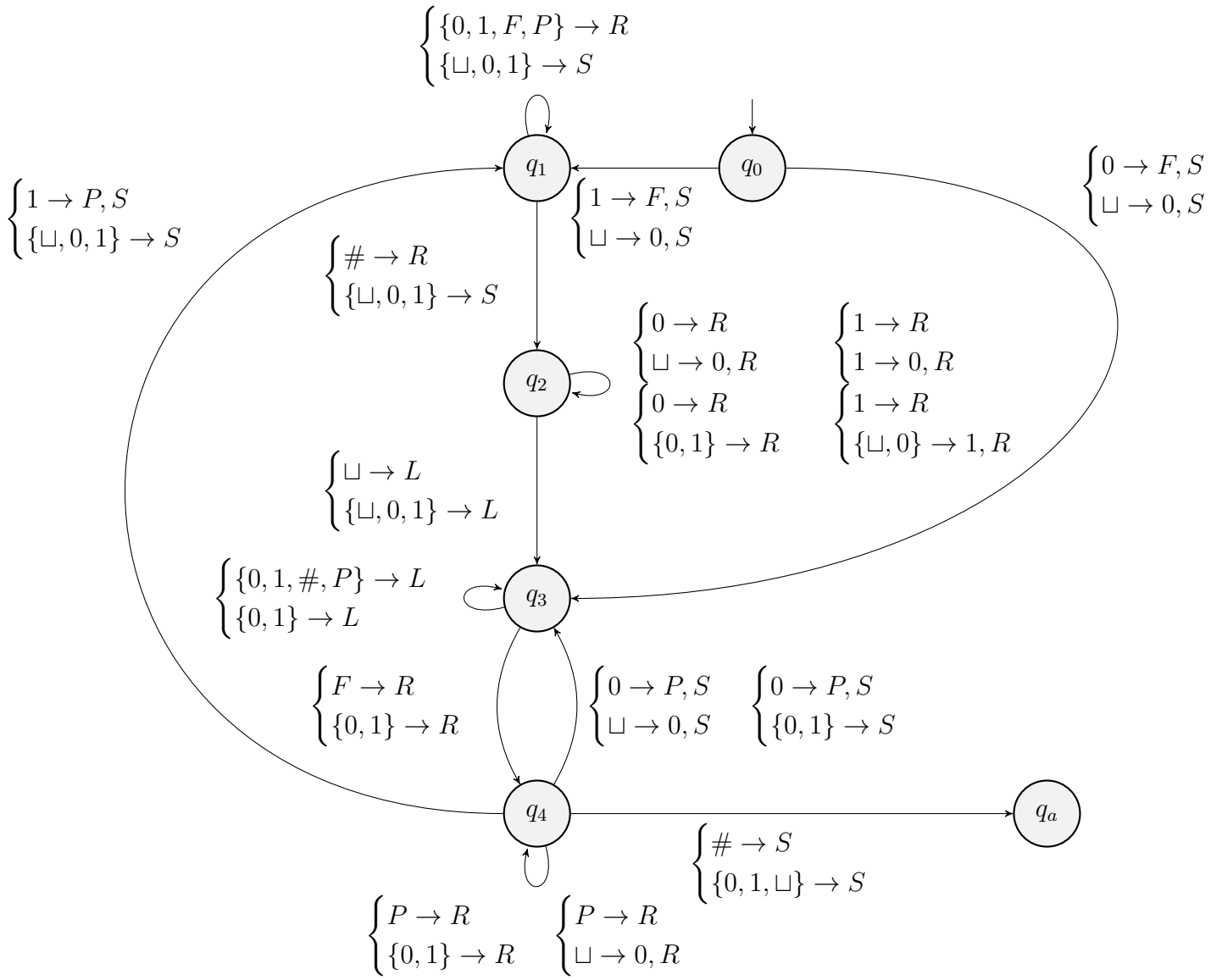
Hint: for handling $a_i b_j$, where $a_i, b_j \in \{0, 1\}$, one link is enough by the binary system property. This may help you to have the # of states within the limit.

(b) (10 pts) Please simulate the machine on a string

$$01\#11.$$

Solution.

(a) Please see the following diagram.



Let us check the relationship between the diagram and our steps.

- In the beginning, we modify a_0 to F via the paths $\overrightarrow{q_0q_1}$ and $\overrightarrow{q_0q_3}$. This process corresponds to Step 1. Moreover, if $a_0 = 0$, we can pass Step 2 and Step 3.
- The paths $\overrightarrow{q_1q_1}$ and $\overrightarrow{q_1q_2}$ are for Step 2.
- In Step 3, we do the multiplication

$$a_i(b_0 + b_1x + \cdots + b_mx^m)$$

  by $\overrightarrow{q_2q_2}$.
  - If $b_j = 0$, then we move both heads to the right without changing the 2nd tape. However, if the 2nd tape head points to $\sqcup$, we must change $\sqcup$ to 0.

– If $b_j = 1$, we need $1 \to 0, R$ in the second tape because $1 + 1 = 0$. Otherwise, the content of the 2nd tape should become 1.

- For Step 4, we use $\overrightarrow{q_2q_3}$, $\overrightarrow{q_3q_3}$ and $\overrightarrow{q_3q_4}$ to go to the first position of both tapes. The loop $\overrightarrow{q_3q_3}$ stops if we see F in the 1st tape ($\overrightarrow{q_3q_4}$).
- We implement Step 5 by $\overrightarrow{q_3q_4}$ and a loop on $\overrightarrow{q_4q_4}$ passing all F and P in the first tape. Once we see 0 or 1 in the 1st tape, we go to Step 1. In this situation, Step 6 is also done because the index update is for the algorithm and the TM does not need it. On the other hand, if we see # in the first tape, all the coefficients $a_0, \ldots, a_n$ have been processed and we can go to q_a. In Step 1, we finish this step by running $\overrightarrow{q_4q_3}$ and $\overrightarrow{q_4q_1}$. Note that we use $\overrightarrow{q_4q_3}$ to skip Step 2 and Step 3 if $a_i = 0$.

Common mistakes:

- In TM, we do not draw a double circle in the accepted state.
- In P169 of textbook, we know that "For the left-hand end, the configuration q_ibv yields q_jcv if the transition is left-moving (because we prevent the machine from going off the left-hand end of the tape)". Some students draw diagrams that conflict to this policy.
- Some students use their own notations without explanation.

(b) Here is the simulation.

$$\begin{bmatrix} q_0 & 0 & 1 & \# & 1 & 1 \\ q_0 & \sqcup & \sqcup & \sqcup & \sqcup & \sqcup \end{bmatrix} \Rightarrow \begin{bmatrix} q_3 & F & 1 & \# & 1 & 1 \\ q_3 & \sqcup & \sqcup & \sqcup & \sqcup & \sqcup \end{bmatrix} \Rightarrow \begin{bmatrix} F & q_4 & 1 & \# & 1 & 1 \\ 0 & q_4 & \sqcup & \sqcup & \sqcup & \sqcup \end{bmatrix}$$

$$\Rightarrow \begin{bmatrix} F & q_1 & P & \# & 1 & 1 \\ 0 & q_1 & \sqcup & \sqcup & \sqcup & \sqcup \end{bmatrix} \Rightarrow \begin{bmatrix} F & P & q_1 & \# & 1 & 1 \\ 0 & q_1 & \sqcup & \sqcup & \sqcup & \sqcup \end{bmatrix} \Rightarrow \begin{bmatrix} F & P & \# & q_2 & 1 & 1 \\ 0 & q_2 & \sqcup & \sqcup & \sqcup & \sqcup \end{bmatrix}$$

$$\Rightarrow \begin{bmatrix} F & P & \# & 1 & q_2 & 1 \\ 0 & 1 & q_2 & \sqcup & \sqcup & \sqcup \end{bmatrix} \Rightarrow \begin{bmatrix} F & P & \# & 1 & 1 & q_2 & \sqcup \\ 0 & 1 & 1 & q_2 & \sqcup & \sqcup & \sqcup \end{bmatrix} \Rightarrow \begin{bmatrix} F & P & \# & 1 & q_3 & 1 & \sqcup \\ 0 & 1 & q_3 & 1 & \sqcup & \sqcup & \sqcup \end{bmatrix}$$

$$\Rightarrow \begin{bmatrix} F & P & \# & q_3 & 1 & 1 & \sqcup \\ 0 & q_3 & 1 & 1 & \sqcup & \sqcup & \sqcup \end{bmatrix} \Rightarrow \begin{bmatrix} F & P & q_3 & \# & 1 & 1 & \sqcup \\ q_3 & 0 & 1 & 1 & \sqcup & \sqcup & \sqcup \end{bmatrix} \Rightarrow \begin{bmatrix} F & q_3 & P & \# & 1 & 1 & \sqcup \\ q_3 & 0 & 1 & 1 & \sqcup & \sqcup & \sqcup \end{bmatrix}$$

$$\Rightarrow \begin{bmatrix} q_3 & F & P & \# & 1 & 1 & \sqcup \\ q_3 & 0 & 1 & 1 & \sqcup & \sqcup & \sqcup \end{bmatrix} \Rightarrow \begin{bmatrix} F & q_4 & P & \# & 1 & 1 & \sqcup \\ 0 & q_4 & 1 & 1 & \sqcup & \sqcup & \sqcup \end{bmatrix} \Rightarrow \begin{bmatrix} F & P & q_4 & \# & 1 & 1 & \sqcup \\ 0 & 1 & q_4 & 1 & \sqcup & \sqcup & \sqcup \end{bmatrix}$$

$\Rightarrow$Terminate.

Problem 3 (20 pts). Let $A = \{\langle M \rangle \mid M \text{ is a DFA that accepts all strings in } (01)^*\}$.

(a) (10 pts) Prove that for a DFA M,

$$M \text{ accepts all strings in } (01)^* \iff \overline{L(M)} \cap (01)^* = \emptyset.$$

You must prove the two directions in a formal way.

(b) (10 pts) Construct a decider H to show that A is decidable.

(Hint: you may use the fact that the language

$$E_{\text{DFA}} = \{\langle M \rangle \mid M \text{ is a DFA with } L(M) = \emptyset\}$$

is decidable.)

Solution.

(a) ($\Rightarrow$): Assume for contradiction that $\overline{L(M)} \cap (01)^* \neq \emptyset$. There exists $x \in \overline{L(M)} \cap (01)^*$. Then $x \in \overline{L(M)}$ and $x \in (01)^*$. That is, x is a string in $(01)^*$ rejected by M, a contradiction.

($\Leftarrow$): Assume for contradiction that M doesn't accepts all strings in $(01)^*$. There exists $x \in (01)^*$ rejected by M. Then $x \in \overline{L(M)} \cap (01)^*$, a contradiction.

(b) Since regular languages are closed under complement and intersection, we can construct a DFA that recognizes $\overline{L(M)} \cap (01)^*$. Also, there exists a decider H that decides E_{DFA}. Therefore, we design the following TM recognizing A.

On input $\langle M \rangle$ where M is a DFA:

1. Construct DFA B that recognizes $\overline{L(M)} \cap (01)^*$.
2. Run decider H on input $\langle B \rangle$.
3. If H accepts, accept. If H rejects, reject.

Problem 4 (30 pts). In machine learning, linear separability is a special property for a given data set, which means we can use a hyper-plane to separate two classes of data instances. For example, suppose we have a data set,

$$D = \{(y_i, \boldsymbol{x}_i) \in \{-1, 1\} \times \mathbb{R}^2 \mid i = 1, \ldots, n\},$$

where $y_i = \pm 1$ is the class label and $\boldsymbol{x}_i \in \mathbb{R}^2$ is a data instance, the set is linearly separable if there exist $\boldsymbol{w}$ and b such that

$$\begin{cases} \boldsymbol{w}^T \boldsymbol{x}_i + b > 0 & \text{if } y_i = 1 \\ \boldsymbol{w}^T \boldsymbol{x}_i + b < 0 & \text{if } y_i = -1 \end{cases}$$

for all i. In the special case of $\boldsymbol{x} \in \mathbb{R}^2$, if the data set contains at least two data points in each class, an equivalent way to describe the linear separability is as follows.

$$\overline{\boldsymbol{x}_i \boldsymbol{x}_j} \cap \overline{\boldsymbol{x}_h \boldsymbol{x}_k} = \emptyset \text{ if } y_i y_j = 1, y_h y_k = 1 \text{ and } y_i y_h = -1,$$

where $\overline{\boldsymbol{x}_i \boldsymbol{x}_j}$ means the line segment between $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$. In other words,

> for any two points $\boldsymbol{x}_i, \boldsymbol{x}_j$ from the same class, and any two other points $\boldsymbol{x}_h, \boldsymbol{x}_k$ from the opposite class, the two segments $\overline{\boldsymbol{x}_i \boldsymbol{x}_j}$ and $\overline{\boldsymbol{x}_h \boldsymbol{x}_k}$ do not overlap. (4)

(a) (5 pts) Consider the data set

$$\left\{(+1, \begin{bmatrix} 1 \\ 1 \end{bmatrix}), (+1, \begin{bmatrix} 2 \\ 3 \end{bmatrix}), (-1, \begin{bmatrix} -2 \\ -1 \end{bmatrix}), (-1, \begin{bmatrix} -1 \\ 2 \end{bmatrix}), (-1, \begin{bmatrix} -1 \\ -1 \end{bmatrix})\right\}.$$

Please check whether the data set is linearly separable by using the description in (4). In this subproblem, you can draw a figure in an $\mathbb{R}^2$ plane and give some explaination as your answer.

(b) (5 pts) If D is not linearly separable, we can remove m data

$$D_m = \{(y_{a_j}, \boldsymbol{x}_{a_j}) \in \{-1, 1\} \times \mathbb{R}^2 \mid a_j \in \{1, \ldots, n\}, j = 1, \ldots, m\}, \tag{5}$$

such that the subset $D \setminus D_m$ is linearly separable. By using the description in (4), please check whether for the data set

$$\hat{D} = \left\{(+1, \begin{bmatrix} -1 \\ 0 \end{bmatrix}), (+1, \begin{bmatrix} 1 \\ 0 \end{bmatrix}), (+1, \begin{bmatrix} 0 \\ -1 \end{bmatrix}), (-1, \begin{bmatrix} 0 \\ 1 \end{bmatrix}), (-1, \begin{bmatrix} 0 \\ 0 \end{bmatrix}), (-1, \begin{bmatrix} 1 \\ -1 \end{bmatrix}), (-1, \begin{bmatrix} 2 \\ -2 \end{bmatrix})\right\}$$

we can remove **one** data such that $\hat{D} \setminus D_1$ is linearly separable. In this subproblem, you can draw a figure in an $\mathbb{R}^2$ plane and give some explaination as your answer.

(c) (10 pts) Now, consider

$$L = \left\{ \langle D, m \rangle \mid D \setminus D_m \text{ contains at least two data in each class, and is linearly separable in } \mathbb{R}^2 \right\}.$$

Please design a polynomial verifier V with the certificate

$$c = D_m, \text{defined in (5), for some } a_1, \ldots, a_m \in \{1, \ldots, n\},$$

to show that $L \in \text{NP}$. Note that

- your verifier algorithm must include clear, step-by-step, high-level descriptions,
- we assume that for given points $A, B, C, D \in \mathbb{R}^2$, checking

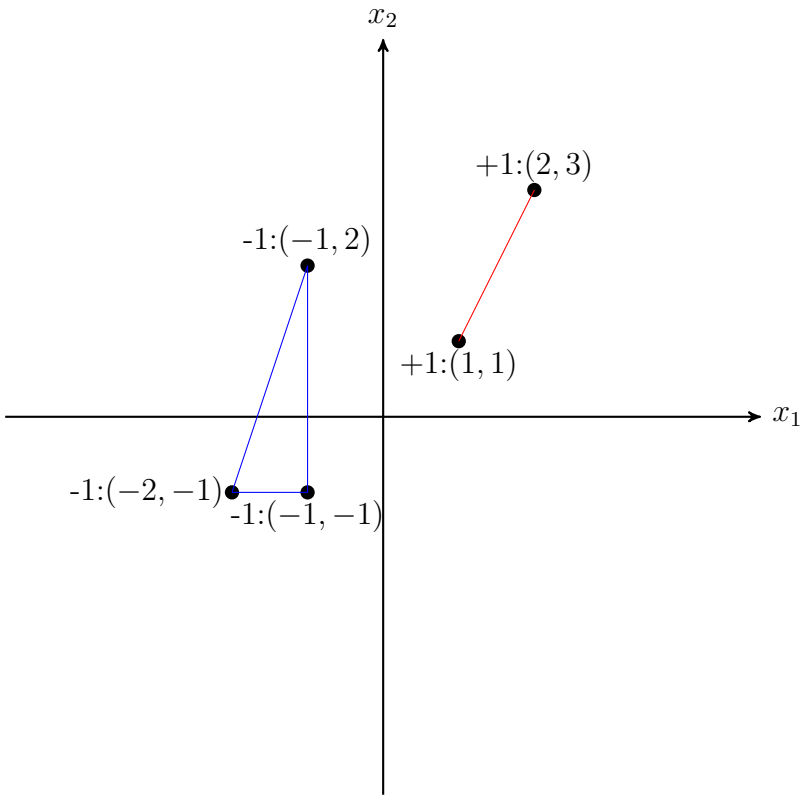
$$\overline{AB} \cap \overline{CD} = \emptyset,$$

  costs a polynomial time $g(n)$ in a TM, and
- you must analyze your algorithm in each step and further show the complexity in a TM.

(d) (10 pts) Show that $L \in \text{NP}$ again by designing an NTM that uses the verifier V in (c) as a subroutine.

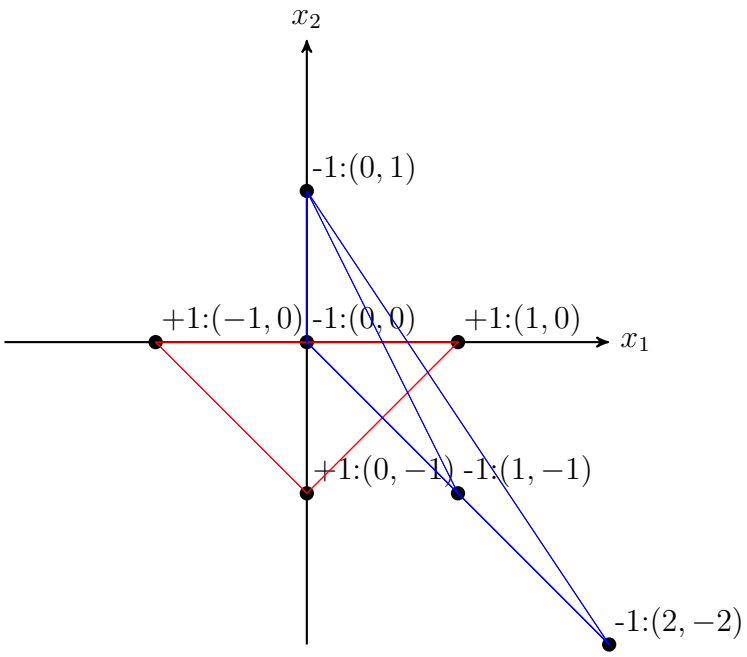
Solution.

(a) In the following figure,

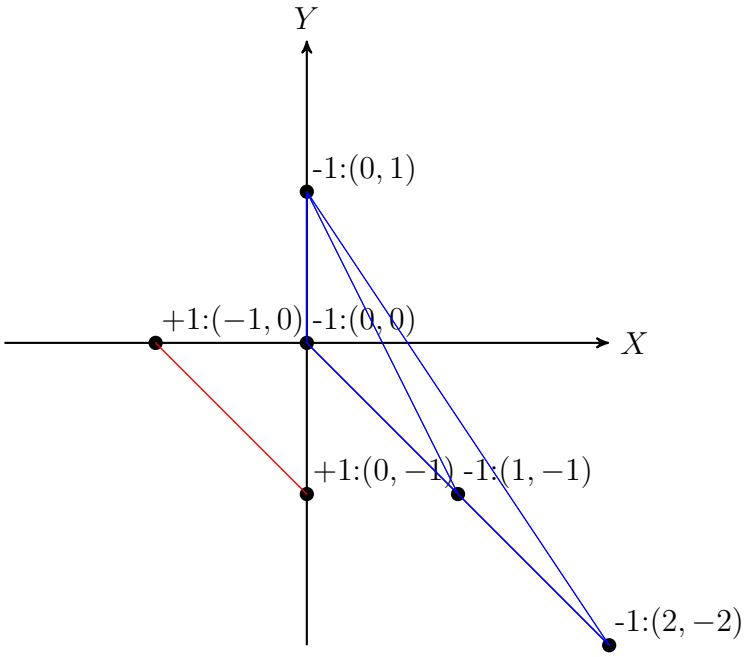


we can see that every blue line does not overlap with the red line. Thus, (4) is staisfied, and this data set is linearly separable.

(b) The following figure shows all the line segments between points within the same class.



By removing the node $(+1, \begin{bmatrix}1\\0\end{bmatrix})$, we can check that the new figure



satisfies (4). Therefore,

$$\hat{D} \setminus \left\{ (+1, \begin{bmatrix}1\\0\end{bmatrix}) \right\}$$

is linearly separable.

(c) Here is the algorithm:

Step-1 Remove the data from D_m, which requires $O(n)$.

Step-2 Check whether there exists at least two data in each class, which requires $O(n)$.

Step-3 Perform a 4-level nested for-loops, two for picking two data in the class "+1", and another two for picking two data in the class "-1", which requires $O(n^4)$. Then, check the overlap between two line segments. Overall, we need $O(n^4 \times g(n))$ in this step.

Therefore, this algorithm requires $O(n^4 \times g(n))$ for verifying the certificate c, which implies $L \in NP$.

(d) We can non-deterministically pick the indices

$$a_1^{(t)}, \ldots, a_m^{(t)}$$

in the certificate for all t in different combinations

$$t = 1, \ldots, \binom{n}{m}.$$

Moreover, we have defined a polynomial time verifier V in (c). Thus, $L \in NP$.