Olympiads in Informatics, 2020, Vol. 14, 21–35
© 2020 *IOI, Vilnius University*
DOI: 10.15388/ioi.2020.02

# Latest Algorithms on Particular Graph Classes

Phan Thuan DO[1,*], Ba Thai PHAM[1,*], Viet Cuong THAN[2]
[1]*Hanoi University of Science and Technology*
[2]*University of Nebraska-Lincoln*
*e-mail: thuandp@soict.hust.edu.vn, thai9cdb@gmail.com, cthan2@huskers.unl.edu*

**Abstract.** Many optimization problems such as Maximum Independent Set, Maximum Clique, Minimum Clique Cover and Maximum Induced Matching are $\mathbb{NP}$-hard on general graphs. However, they could be solved in polynomial time when restricted to some particular graph classes such as comparability and co-comparability graph classes. In this paper, we summarize the latest algorithms solving some classical NP-hard problems on some graph classes over the years. Moreover, we apply the $\alpha$-redundant technique to obtain linear time $\mathcal{O}(|V|)$ algorithms which find a Maximum Induced Matching on interval and circular-arc graphs. Inspired of these results, we have proposed some competitive programming problems for some programming contests in Vietnam in recent years.

**Keywords:** graph classes, interval graph, circular-arc graph, co-comparability, maximum induced matching.

## 1. Introduction

Despite the fact that the study field of particular graph classes has been skyrocketing over the years, there is still modest number of competitive programming problems inspired by this field for Olympic programming contests, especially for high school contests. One reason is that the particular graph classes are excluded in the IOI syllabus. However, many particular properties of these graph classes can be expressed as some other knowledge included in the syllabus. We have explored some of these properties to propose problems for some programming contests in Vietnam in recent years. These problems are not mentioned about particular graph classes by being stated as practical situations. They can be consequently solved by knowledge included in the IOI syllabus.

Usually, $\mathbb{NP}$-hard problems may become much easier when restricted on particular graph classes. Exploring this way, we first study graph optimization problems such as Maximum Independent Set, Maximum Matching, Maximum Clique, Minimum Clique

* Corresponding authors

Cover. We summarize latest algorithms for these problems on some co-comparability graph classes such as interval, permutation and trapezoid graphs.

Besides, in the literature, there exist only a few trivial algorithms of finding Maximum Induced Matching (MIM) on particular graph classes. This problem was first proposed in 1989 by Cameron (Cameron, 1989). While the maximum matching problem can be solved in polynomial time in an arbitrary graph, the MIM problem is a $\mathbb{NP}$-Hard problem, even for bipartite graphs. The MIM problem can be trivially solved in polynomial time for interval graphs and chordal graphs (Cameron, 1989), circular-arc graphs (Golumbic, 1993), Trapezoid graphs and co-comparability graphs (Golumbic and Lewenstein, 2000), Asteroidal-triple-free graphs (Cameron, 2004; Chang, 2001), Weakly chordal graphs (Cameron *et al.*, 2003), Interval-filament graphs (Cameron, 2004). There are algorithms that can find MIM in linear time $\mathcal{O}(|E| + |V|)$ in chordal graphs (Brandstädt and Hoàng, 2008), interval graphs (Golumbic and Lewenstein, 2000), tree graphs (Golumbic and Lewenstein, 2000) and permutation bipartite graphs (Chang, 2001). In this paper, we present two $\mathcal{O}(|V|)$ algorithm solving the MIM problem on interval and circular-arc graphs. Our approach is to use the $\alpha$-redundant technique to reduce the search space while still preserving optimal solutions.

In the last section, we describe some of our competitive programming problems proposed for Vietnam Team Selection Tests (TST) in recent years. These problems are inspired by the latest results on circular-arc, trapezoid and disc graphs for the problems of maximum induced matching and minimum vertex cover.

## 2. Preliminary

### 2.1. $\mathbb{NP}$-*Hard Graph Problems*

Given a graph $G = (V, E)$, a set $S \subseteq V$ is called an independent set of $G$ if no two members of $S$ are adjacent. The number of vertices in a maximum independent set (MIS) of $G$ is called the independence number, denoted by $\alpha(G)$.

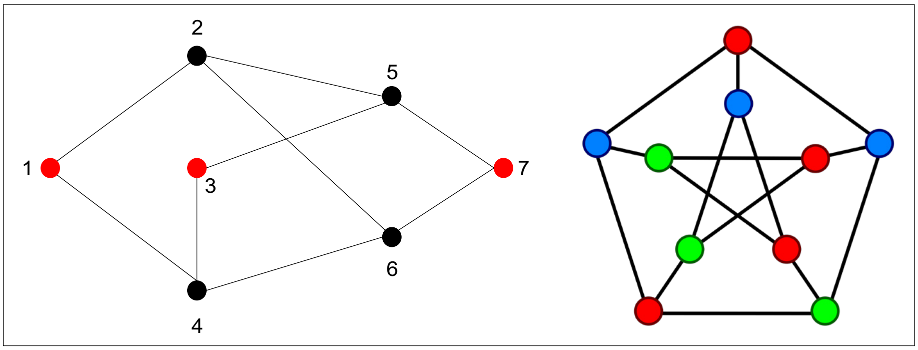

Fig. 1. An illustration of Independent Set and Graph Coloring.

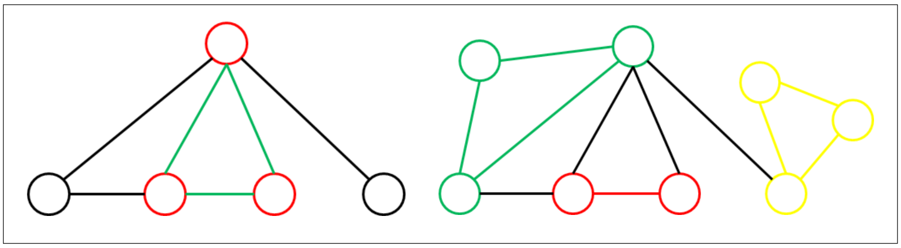

Fig. 2. An illustration of Clique and Clique Cover.

A subset $C$ of $V$ is a clique if and only if every two vertices of $C$ are adjacent. A clique of graph $G$ corresponds to an independent set of the complement graph $\overline{G}$. The cardinality of a maximum cardinality clique (MC) is called $\omega(G)$. A clique cover of $G$ is a partition of the set $V$ into cliques. The number of cliques in a minimum clique (MCC) cover of $G$ is denoted by $k(G)$.

A coloring of a graph is an assignment of labels, also called color, to each vertex such that no two adjacent vertices share the same color. The minimum number of colors needed to assign a graph subject the constraint is called the chromatic number of that graph and is denoted by $\chi(G)$. Vertices with the same color of $G$ are in a clique of the complement graph $\overline{G}$ of $G$. Hence a coloring of $G$ is a minimum clique cover of $\overline{G}$.

These four problems MIS, MC, MCC, and Coloring have been known to be $\mathbb{NP}$ – hard in general graphs. However, many of them can be solved efficiently with polynomial time complexity in following particular graph classes.

## 2.2. *Particular Graph Classes*

A graph $G$ is comparability if there is a transitive orientation, an assignment of directions to the edges of the graph, *i.e.* an orientation of the graph, such that the adjacency relation of the resulting directed graph is transitive: whenever there exist directed edges $(x, y)$ and $(y, z)$, there must exist an edge $(x, z)$. A co-comparability graph is a complement of the comparability graph. The MC and Coloring problem in comparability graphs can be solved in linear time $\mathcal{O}(|E| + |V|)$ using the lexicographic depth-first search algorithm (Golumbic, 2004) while a maximum independent set and minimum clique cover could be found by using maximum flow algorithms (Golumbic, 2004). These algorithms could be taken advantage of solving MIS, MC, MCC and Coloring problems in co-comparability graphs.

A graph is an intersection graph if each vertex corresponds to a set and two vertices are adjacent iff their set share same members. If each set is an interval on a line, the graph is called an interval graph.

A simple greedy $\mathcal{O}(|V|)$ algorithm can be used to solve the MIS and MCC problem in interval graphs, with an assumption that every interval in the input is sorted by their left ends. The Coloring and MC problem can be solved in $\mathcal{O}(|V| \log |V|)$.

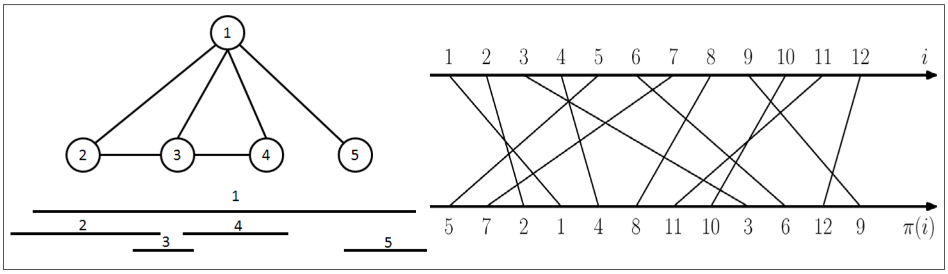
1
2
3
4
5
1
2
4
3
5
1 2 3 4 5 6 7 8 9 10 11 12 $i$
5 7 2 1 4 8 11 10 3 6 12 9 $\pi(i)$

Fig. 3. An illustration of Interval Graph and Permutation Graph.

A permutation graph is a graph whose vertices represent the elements of a permutation, and whose edges represent pairs of elements that are reversed by the permutation. Permutation graphs may also be defined geometrically, as the intersection graphs of line segments whose endpoints lie on two parallel lines.

Permutation graphs are both comparability and co-comparability. There is an $\mathcal{O}(|V|\log\log|V|)$ algorithm based on the longest increasing subsequence to solve the MIS problem in permutation graphs. This algorithm can be used to find MC, MCC and Coloring also in $\mathcal{O}(|V|\log\log|V|)$.

A trapezoid graph is an intersection graph of trapezoids in which two sides line on two parallel lines. The MIS and MCC problem in this graph class can be solved in $\mathcal{O}(|V|\log\log|V|)$ using the sweep line technique (Felsner *et al.*, 1997). A MC and Coloring could be found in $\mathcal{O}(|V|\log|V|)$.

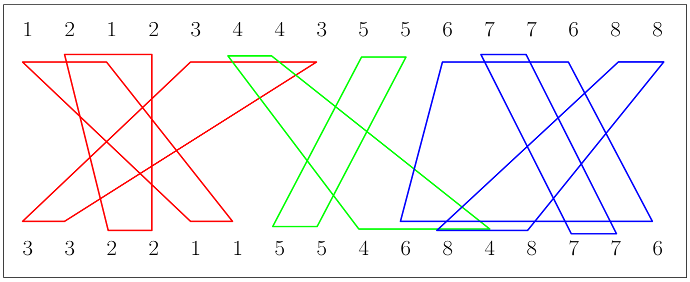

Fig. 4. An illustration of Trapezoid Graph.

Table 1
Latest algorithms on some particular graph classes

| Problem | Comparability | Co-comparability | Interval | Permutation | Trapezoid |
|---|---|---|---|---|---|
| MC | $\mathcal{O}(\|E\|+\|V\|)$ | Polynomial | $\mathcal{O}(\|V\|\log\|V\|)$ | $\mathcal{O}(\|V\|\log\log\|V\|)$ | $\mathcal{O}(\|V\|\log\|V\|)$ |
| Coloring | $\mathcal{O}(\|E\|+\|V\|)$ | Polynomial | $\mathcal{O}(\|V\|\log\|V\|)$ | $\mathcal{O}(\|V\|\log\log\|V\|)$ | $\mathcal{O}(\|V\|\log\|V\|)$ |
| MIS | Polynomial | $\mathcal{O}(\|E\|+\|V\|)$ | $\mathcal{O}(\|V\|)$ | $\mathcal{O}(\|V\|\log\log\|V\|)$ | $\mathcal{O}(\|V\|\log\log\|V\|)$ |
| MCC | Polynomial | $\mathcal{O}(\|E\|+\|V\|)$ | $\mathcal{O}(\|V\|)$ | $\mathcal{O}(\|V\|\log\log\|V\|)$ | $\mathcal{O}(\|V\|\log\log\|V\|)$ |

## 3. The $\alpha$ - Redundant Method

Given a graph $G = (V, E)$, the $k$-th power graph of $G$ is denoted by $G^k = (V, E')$ having the same vertex set with $G$. Two vertices $u$, $v$ in $G^k$ are adjacent iff there exists a path from $u$ to $v$ of length less than or equal to $k$. Let $L(G)$ denote the line graph of $G$, *i.e.*, each edge of $G$ is a vertex of $L(G)$, two vertices of $L(G)$ are adjacent iff two corresponding edges share a common endpoint. An edge set $M \subseteq E$ is called a matching of $G$ iff there does not exist a pair of edges in $M$ with a common vertex. An induced matching of $G$ is a matching where the distance between two arbitrary vertices in two different edges is at least two.

An induced matching of a graph $G$ corresponds with an independent set of $L(G)^2$. So there will be a polynomial complexity algorithm for MIM whenever MIS of a graph can be found in polynomial time. In some circumstances, avoiding fully constructing the graph $L(G)^2$ may lead to better time complexity.

A vertex of $G$ is $\alpha$ - redundant iff its removal does not affect the size of the MIS in $G$. The approach is to remove $\alpha$ - redundant vertices from $L^2(G)$ before finding a MIS of the remaining graph $L^*$.

## 4. Maximum Induced Matching in Interval Graphs

In this section, we propose a linear time $\mathcal{O}(|V|)$ algorithm solving the MIM problem in interval graphs based on the $\alpha$-redundant technique with an assumption that every interval in the input is sorted by their left ends. The algorithm will be improved to find a MIM in circular-arc graphs with the same computational complexity $\mathcal{O}(|V|)$. For any interval $u$, we denote the coordinate of its left and right end by $l(u)$ and $r(u)$, respectively. We first give an important property.

**Lemma 1.** *If $G$ is an interval graph, the graph $L^2(G)$ is also an interval graph.*

*Proof.* For each edge $(u, v)$, with $l(u) \le l(v)$, we draw a line from $l(u)$ to $\max\{r(u), r(v)\}$, the right end of the union line of $u$ and $v$. $L^2(G)$ is an interval graph in which

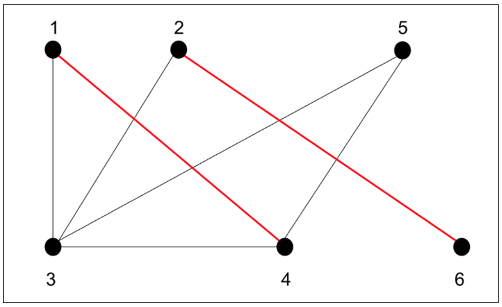

Fig. 5. An illustration of induced matching.

each vertex of $L^2(G)$ corresponding with its union line. If $(u, v)$ and $(w, z)$ are adjacent, there is at least one edge between 2 two pairs of vertices $(u, v)$ and $(w, z)$. Assume that the interval $u$ and $w$ cut each other. So the union interval $(u, v)$ and $(w, z)$ also intersect each other. It is trivial that if two union interval $(u, v)$ and $(w, z)$ are not adjacent, there does not exist any edge connect $u$ or $v$ to one of $\{w, z\}$.

### 4.1. *The Construction of* $L^*$

From the aforementioned lemma, the graph $L^2(G)$ is an interval graph so the new graph $L^*$ is an interval graph as well. Following the same idea in finding a MIM in permutation graphs, we will remove $\alpha$ - redundant vertices from $L^2(G)$. For each interval $u$, the algorithm will find the *optimal* interval $v$ of $u$. The definition of the optimal interval can be express as follows:

**Definition 1.** Given an interval $u$, the optimal interval $v$ of $u$ is interval that $r(u) \geq l(v) \geq l(u)$ and union interval $(u, v)$ is shortest.

If there exists an optimal interval $v$ of interval $u$, every union interval of $u$ and an interval $v$ 'smaller' than $u$ except $(u, v)$ is $\alpha$ - redundant. This property can be proved by the argument exchange method because every interval in the maximum independent set of $L^2(G)$ can be replaced by a union interval of a vertex and its optimal interval. So the number of vertices in $L^*$ is at most $n$ and finding the maximum independent set of an interval graph costs $\mathcal{O}(|V|)$ if the intervals are sorted by their left ends.

### 4.2. *Algorithm*

The algorithms finding a MIM of an interval graph can be divided into two steps, the first one is constructing the graph $L^*$ and the second is finding a MIS of $L^*$. The intervals in the input are already sorted by their left ends.

**Algorithm 1** $Optimal(V)$

```
1:  Input: Set of intervals $V$.
2:  Output: Optimal interval of each members in $V$
3:  $optimal\_vertex(u) \leftarrow null$ for every $u \in V$
4:  Stack $S$ $\emptyset$
5:  for $u \in V$ do
6:      while $S = \emptyset$ do
7:          $v \leftarrow S.top$
8:          if $r(v) < r(u)$ then
9:              Break
10:         end if
11:         $optimal\_vertex(v) \leftarrow u$
12:         $S.pop$
```

```
13:     end while
14:     if $S \neq \emptyset$ and $l(u) \leq r(S.top)$ then
15:         $optimal\_vertex(S.top) \leftarrow u$
16:     end if
17:     $S.push(u)$
18: end for
19: Return $optimal\_vertex$
```

**Algorithm 2** *$MIM\ Interval(V)$*

```
1: Input: Set of intervals $V$
2: Output: An MIM of the interval graph $G(V)$
3: $optimal\_vertex \leftarrow Optimal(V)$
4: $V^* \leftarrow \emptyset$
5: for $u \in V$ do
6:     if $optimal\_vertex(u) \neq null$ then
7:         $v \leftarrow optimal\_vertex(u)$
8:         Add the interval $u \cup v$
9:     end if
10: end for
11: Stack $S \leftarrow \emptyset$
12: for $u \in V^*$ do
13:     if $S = \emptyset$ then
14:         $v \leftarrow S.top$
15:         if $v$ does not cut $u$ then
16:             $S$.push($v$)
17:         end if
18:     end if
19: end for
20: Return $S$
```

### 4.3. *Complexity*

Each interval in $V$ will be pushed and popped at most once. So, the time complexity of the algorithm is $\mathcal{O}(|V|)$

## 5. MIM in Circular-arc Graphs

The algorithm finding a MIM in interval graphs can be extended to solve the MIM problem in circular-arc graphs. Like in interval graphs, we give first the important property of $L^2(G)$ of a circular-arc graph $G$.

**Lemma 2.** *The graph $L^2(G)$ of a circular-arc graph $G$ is also a circular-arc graph.*

This lemma can be proven using the same technique as in interval graphs. Assume that there is an origin in the circle, every end of the arc are coordinated. $l$ is the left end of an arc $(l, r)$ if and only if the arc from $l$ to $r$ is clockwise. We propose an $\mathcal{O}(|V|)$ algorithm to find a MIM of a circular-arc graph with the assumption that the arcs are already sorted by their left end and their length. The idea of the algorithm is always to find an optimal neighbor corresponding with each vertex. We first re-coordinate the arcs to identify the stating arc on the circular, and then applying the same algorithm for interval graphs from the starting arc.

**Algorithm 3** $Optimal(V)$

```
1: Input: $V$ is a sorted set of arcs.
2: Output: Optimal neighbor with each vertex of $V$.
3: $u0$ is the shortest arc of $V$
4: Re-coordinate arc in $V$ with the origin is the left end of $u_0$.
5: Stack $S \leftarrow \emptyset$
6: $optimal\ vertex(u) \leftarrow null$ for every $u \in V$
7: for $u \in V$ do
8:     while $S \neq \emptyset$ do
9:         $v \leftarrow S.top$
10:        if $r(v) < r(u)$ then
11:            Break
12:        end if
13:        $optimal\ vertex(v) \leftarrow u$
14:        $S$.pop
15:    end while
16:    if $S \neq \emptyset$ and $l(u) \leq r(S.top)$ then
17:        $optimal\ vertex(S.top) \leftarrow u$
18:    end if
19:    $S$.push($u$)
20: end for
21: Update $optimal\ vertex$ of arcs intersecting $u_0$
22: Return $optimal\ vertex$
```

The correctness of the algorithm is proven by the following lemma.

**Lemma 3.** *The procedure $Optimal(V)$ return the optimal arc of each arc in $V$.*

*Proof.* With an arc $u$, if $r(u) \geq l(u)$, the work finding its optimal neighbor is similar to interval graph. Consider the arc $u$ with $r(u) < l(u)$, which means this arc contains the origin, we will prove that the optimal neighbor of $u$ can only be an arc $v$ that $r(v) < l(v)$, or be $u_0$. If the optimal neighbor of $u$ is an arc $w$ that $r(w) > r(u)$ and $r(w) > l(w)$, which mean this arc does not contain the origin, the length of the union arc $(u, w)$ is smaller than that of $(u, u_0)$. So, $r(w) < r(u_0)$ and $l(w) < l(u_0)$. This implies length of $w$ is smaller than length of $u_0$, which is contrary to our choice that $u_0$ is the shortest arc.

## 6. Some Induced Competitive Programming Problems

### 6.1. *Circular-arc (Vietnam TST 2018)*

Given a circle defined by the center coordinates $(x, y)$, radius $r$ and $n$ lines, in which the $i^{th}$ line is determined by the equation $a_i x + b_i y + c_i = 0$. There is no secant passing through the center of the circle. A secant if cut the circle at $2$ points $A$ and $B$, the arc $AB$ small of the circle is called the characteristic arc of that line. Note that if the secant touches the circle, the characteristic arc degrades to $1$ point. Next, to examine the relationship between arcs, Hai constructs a simple undirected graph $G = (V, E)$, where each vertex of $V$ corresponds to a typical arc of the circle, and the set edge $E$ consists of all the edges connecting the two vertices in $V$ where the two characteristic arcs correspond to them. We call the path length $d$ between two edges $e$ and $f$ as a sequence of edges $(e, g_1, g_2, ..., g_d, f)$ so that two consecutive edges in this sequence have a common vertex. The distance between the two sides $e$ and $f$ is the length of the shortest path between them. If there is no path between $e$ and $f$, the distance between them is set to $+\inf$. Hai wants to find the set of edges $E' \subseteq E$ with the largest cardinality such that the distance between any two edges in $E'$ is at least 2. See Fig. 6 for an example.

The problem can be directly solved by the $\mathcal{O}(n)$ MIM algorithm on Section 5 plus the time $\mathcal{O}(n \log n)$ to sort arcs by their left end and their length. Hence the size $n$ of input can go to $10^6$.

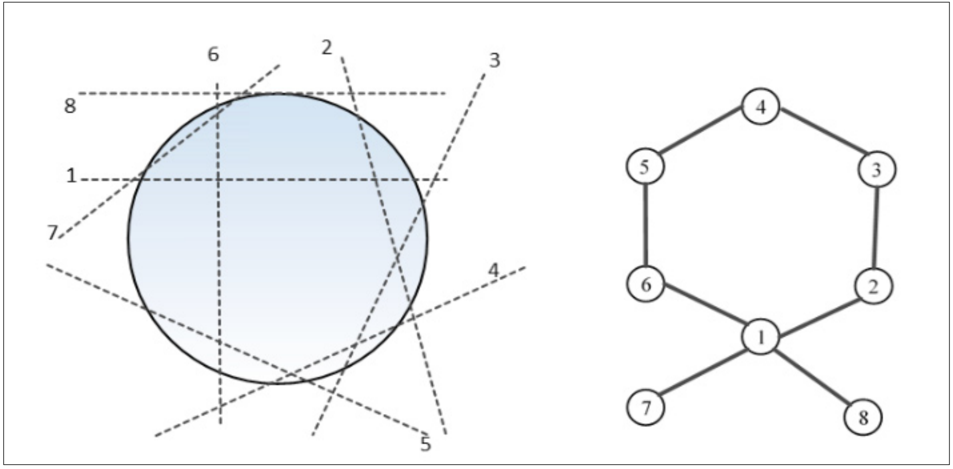

Fig. 6. An example of Problem 6.1.

6.2. *ESEA (Vietnam TST Camp 2018)*

The eastern territorial sea has $n$ critical area ($n \leq 10^6$). The entire territorial sea is depicted on a map of coordinates, where each critical area is represented by a rectangle with vertices at points with integer coordinates. In preparation for the unprecedented training session "ESEA" at sea, the Naval Military Command is planning a battle on the original map of simulated territorial waters. $n$ pair of detectors, each pair of detectors ($\delta_1$, $\delta_2$) at two critical points:

- $\delta_1$ detector set at coordinates ($x_1$, $y_1$) is capable of detecting objects within its left lower quadrant, *i.e.* all points with coordinates ($u$, $v$) satisfies: $u \leq x_1$ and $v \leq y_1$.
- $\delta_2$ detector set at coordinates ($x_2$, $y_2$) is capable of detecting objects within its right higher quadrant, i.e. all points with coordinates ($u$, $v$) satisfies: $u \geq x_2$ and $v \geq y_2$. Know that $x_1 \leq x_2,\ y_1 \leq y_2$.

Two pairs of detectors $i$ and $j$ are called **interconnected** if both detectors of $j$ pair are fully within the detection range of either detector of $i$.

The military command requires the collection of sets of detectors into at least groups so that each pair must belong to exactly one group and in each group, there are no two pairs that are interconnected. See Fig. 7 for an example.

*Hint.* Consider two parallel lines $X$ and $Y$ . Each point $(x_0, y_0)$ corresponds with a line connecting the point $x_0$ on $X$ with $y_0$ on $Y$. Each detector $(\delta_1, \delta_2)$ forms a trapezoid. See Fig. 8 or an Illustration. Two detectors are interconnected iff two trapezoids are separate. The problem becomes to find the MCC of the trapezoid graph constructed from the trapezoid model corresponding to the detectors.

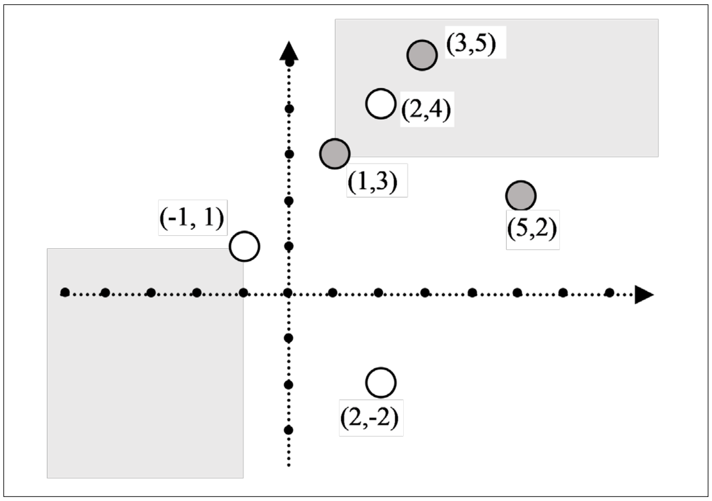

Fig. 7. An example Problem 6.2.

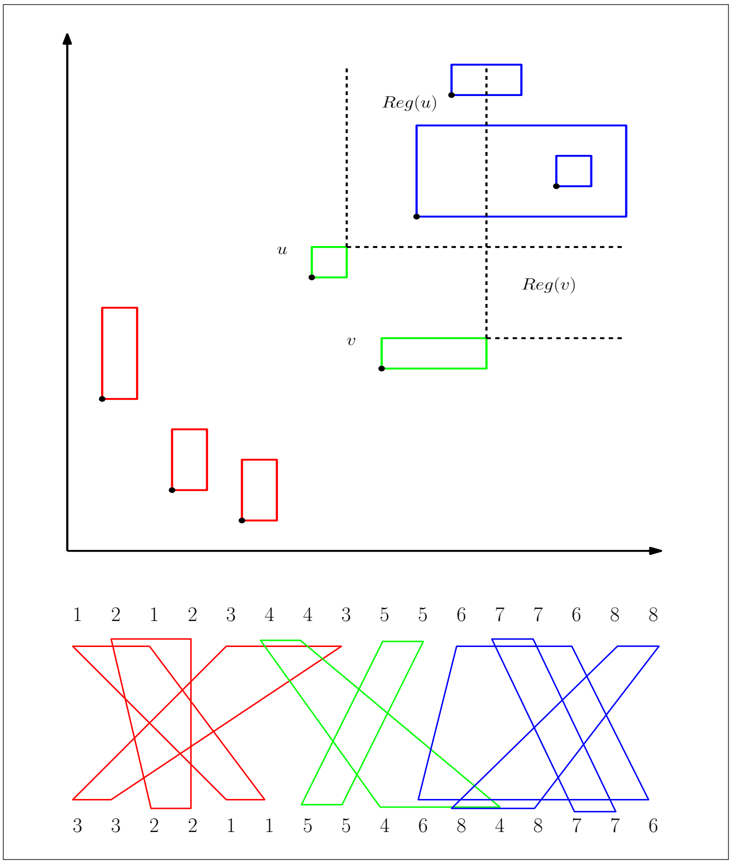

Fig. 8. A Trapezoid Diagram.

We first sort the trapezoids by their left most point. Denoted the cardinality of the maximum independent set of the trapezoid graph $1, 2, \ldots, i$ containing $i$ by $f_i$. Let $F = \max_{i=1}^{n} f_i$, then the partition $S_k = \{i \in \{1, 2, \ldots, n\} : f_i = k\}$, $k = 1, 2, \ldots, F$ is the MCC. Consider two trapezoid $i$, $j$ such that $i < j$, if $i$ and $j$ are not adjacent then $f_j \geq f_i + 1$, so $i$ and $j$ are in 2 different subset of $S$. Therefore $S$ is a clique cover.

Let $P$ be an arbitrary clique cover and $IS$ be an arbitrary independent set. Since any two members of $IS$ must be in 2 different clique then $|P| \geq |IS|$. Otherwise there exists an independent set with size $f_i$ for all $i = 1, 2, \ldots, n$, then $|P| \geq \max_{i=1}^{n} f_i$, in other words $|P| \geq |S|$.

We have a recurrence equation: $f_i = \max_{j<i}^{i \cap j = \emptyset} f_j + 1$. Using the same technique as in finding a longest increasing subsequence, we obtain the computational complexity $\mathcal{O}(n \log n)$ with Binary Indexed Tree.

### 6.3. *The Battle on the River (Vietnam TST 2019)*

Hung is simulating a battle on the river as follows. The map of the river is shown on the coordinate plane. The two banks of the river are given by two parallel lines $x = a$ and $x = b$. There are $n$ piles (numbered from $1$ to $n$) is nailed on the river section, pile $i$ is nailed at the point of integer coordinates $(x_i, y_i)$. Let $c$ and $d$ be the largest and the smallest ordinate respectively. To simplify the problem, each boat battle is considered a circle of diameter $R$. Thus, a boat when entering between two piles $A = (x_A, y_A)$, $B = (x_B, y_B)$ will be stuck if its diameter is larger than the distance between points $A$ and $B$. A boat can cross the river section if it finds a way to move from one point of the river with the ordinate $c + R$ passing through the piles without stuck to reach any point with ordinate $d - R$. Finding the largest value of $R$ so that Hung can cross the river. Constraint $n \leq 10^5$.

*Hint.* Consider each pile is a circle with radius $r$. We can construct an intersection graph $G(r)$ with each vertex corresponding to a circle or a bank. Our problem can be reduced to finding the largest value of $r$ such that two banks are not in the same connected component.

### 6.4. *Building (VOI 2020)*

There are $n$ buildings in Alice's city. In the Cartesian coordinate, a building is represented by a rectangle with sides parallel to the coordinate axes. Two buildings are adjacent if the intersection of their sides is not empty. There is a short path between each

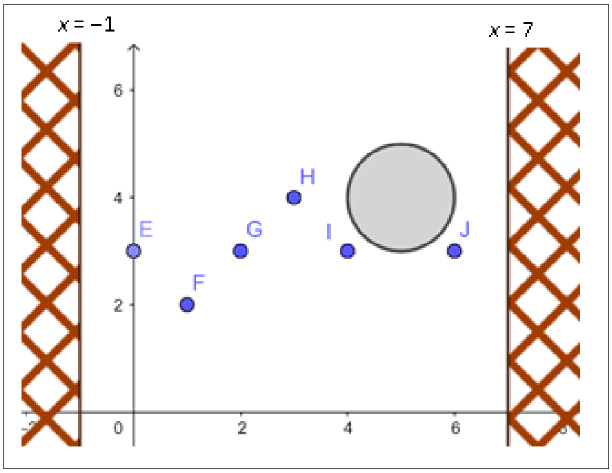

Fig. 9. An example of Problem 6.3.

pair of adjacent buildings. Alice really likes the architectures of the buildings in the city and she usually walks along those short paths. After a few days, she relies some paths are unique-paths. A path $e$ between two building $u$ and $v$ is unique-path if after going from $u$ to $v$, there is no way to come back to $u$ without walking through $e$ again. For each pair of buildings $(u, v)$ which is a unique-path, Alice calculates the maximum number of buildings she can visit for $u$ and for $v$ with the assumption that the path $(u, v)$ is closed, we call those numbers $d_u$ and $d_v$ respectively.

Given the coordinate of each rectangle, help Alice find the pair $(u, v)$ which has an unidirectional path between them and the absolute difference of $d_u$ and $d_v$ is minimum. See Fig.10 for an example.

*Hint.* The problem is related to interval graphs.

1. Construct the graph $G$ representing the adjacent relation between buildings:
   (a) Sort all rectangles in a list by the ascent order of $x$ - coordinate;
   (b) Sort all rectangles in another list by the ascent order of $y$ - coordinate;
   (c) For each rectangle, find its adjacent list by the two sorted lists above. The number of edges in $G$ is only a linear function of the number of rectangles.
4. Use Tarjan's algorithm to find all bridges.
5. Find the bridge $(u, v)$ with minimum absolute difference between $d_u$ and $d_v$. The time complexity is $\mathcal{O}(n \log n)$.

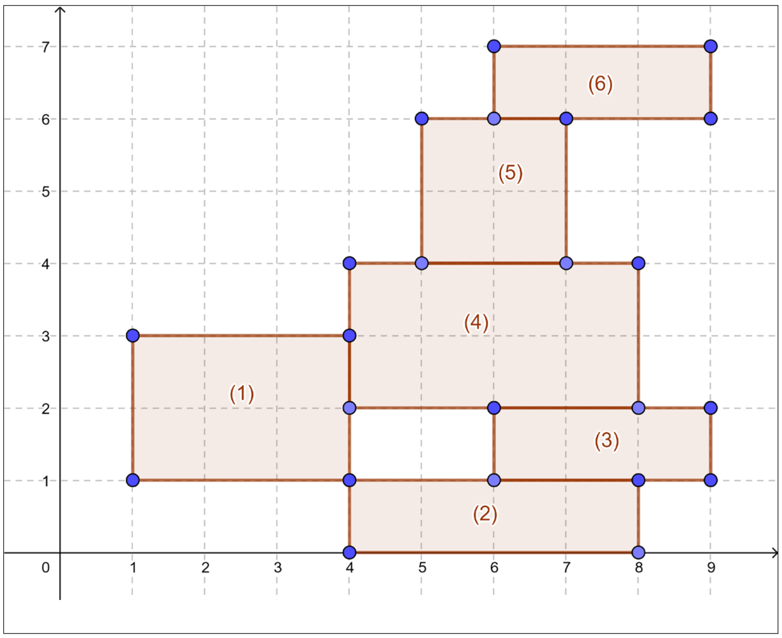

Fig. 10. An example of Problem 6.4.

## 7. Conclusion

Although these induced problems are related to particular graph classes that are excluded in the IOI syllabus, our proposed algorithms to solve these problems are in the scope of the syllabus.

Throughout the paper, our approach of *removing the $\alpha$-redundant vertices* is proved to be effective to reduce the complexity of the algorithms for some problems on particular graph classes. In the future, we intend to apply this method for other suitable problems on some particular cases of graph theory.

## Acknowledgment

This research is funded by the Hanoi University of Science and Technology (HUST) under the project name “Exploiting discrete structures and artificial intelligence to solve optimization problems on mobile IoT networks” with grant number T2020-PC-007. This manuscript was almost accomplished when the first author was working at the Vietnam Institute for Advanced Study in Mathematics (VIASM).

## References

Brandstädt, A. and Hoàng, C. T. (2008). Maximum induced matchings for chordal graphs in linear time. *Algorithmica*, 52(4), 440–447.

Cameron, K. (1989). Induced matchings. *Discrete Applied Mathematics*, 24, 97–102.

Cameron, K. (2004). Induced matchings in intersection graphs. *Discrete Mathematics*, 278, 1–9.

Cameron, K., Sritharanb, R., and Tangb, Y. (2003). Finding a maximum induced matching in weakly chordal graphs. *Discrete Mathematics*, 266, 133–142.

Chang, J. M. (2001). Induced matchings in asteroidal triple free graphs. *Discrete Applied Mathematics*, 132, 67–78.

Felsner, S., Muller, R., and Wernisch, L. (1997). Trapezoid graphs and generalizations, geometry and algorithms. *Cornell Family Papers*.

Golumbic, M. C. (1993). Irredundancy in circular arc graphs. *Discrete Applied Mathematics 4*, pages 79–89.

Golumbic, M. C. (2004). *Algorithmic Graph Theory and Perfect Graphs*, Volume 57. Elsevier.

Golumbic, M. C. and Lewenstein, M. (2000). New results on induced matchings. *Discrete Applied Mathematics*, 101, 157–165.

**P.T. Do** is currently Associate Professor and Deputy-Head of Department of Computer Science at Hanoi University of Science and Technology. He is also a member of the national committee for selecting, training and leading Vietnamese IOI/APIO/ICPC Teams. His current research interests include combinatorics, theory of graphs and applied algorithms in various practical problems such as logistics, network, artificial intelligence and bioinformatics.

**B.T. Pham** is currently a last year student of the talented engineer program at Hanoi University of Science and Technology. He participated in many *ICPC Asia Pacific Regional Contests* during his 5 academic years. He had some publications about graph theory.

**V.C. Than** is currently a Master student at University of Nebraska-Lincoln. He participated in *ICPC Asia Pacific Regional Contests* and in *ICPC North Central North American Regional Contest*. He had some publications about graph theory and game theory.