

# Romeo and Juliet Meeting in Forest like Regions

Neeldhara Misra ✉ 🏠 

Indian Institute of Technology, Gandhinagar, India

Manas Mulpuri ✉

Indian Institute of Technology, Gandhinagar, India

Prafullkumar Tale ✉ 🏠

Indian Institute of Science Education and Research, Pune, India

Gaurav Viramgami ✉

Indian Institute of Technology, Gandhinagar, India

---

## Abstract

The game of rendezvous with adversaries is a game on a graph played by two players: *Facilitator* and *Divider*. Facilitator has two agents and Divider has a team of  $k \geq 1$  agents. While the initial positions of Facilitator's agents are fixed, Divider gets to select the initial positions of his agents. Then, they take turns to move their agents to adjacent vertices (or stay put) with Facilitator's goal to bring both her agents at same vertex and Divider's goal to prevent it. The computational question of interest is to determine if Facilitator has a winning strategy against Divider with  $k$  agents. Fomin, Golovach, and Thilikos [WG, 2021] introduced this game and proved that it is PSPACE-hard and co-W[2]-hard parameterized by the number of agents.

This hardness naturally motivates the structural parameterization of the problem. The authors proved that it admits an FPT algorithm when parameterized by the modular width and the number of allowed rounds. However, they left open the complexity of the problem from the perspective of other structural parameters. In particular, they explicitly asked whether the problem admits an FPT or XP-algorithm with respect to the treewidth of the input graph. We answer this question in the negative and show that RENDEZVOUS is co-NP-hard even for graphs of constant treewidth. Further, we show that the problem is co-W[1]-hard when parameterized by the feedback vertex set number and the number of agents, and is unlikely to admit a polynomial kernel when parameterized by the vertex cover number and the number of agents. Complementing these hardness results, we show that the RENDEZVOUS is FPT when parameterized by both the vertex cover number and the solution size. Finally, for graphs of treewidth at most two and grids, we show that the problem can be solved in polynomial time.

**2012 ACM Subject Classification** Mathematics of computing → Discrete mathematics; Theory of computation → Design and analysis of algorithms

**Keywords and phrases** Games on Graphs, Dynamic Separators, W[1]-hardness, Structural Parameterization, Treewidth

**Digital Object Identifier** 10.4230/LIPIcs.FSTTCS.2022.27

**Related Version** *Full Version:* <https://arxiv.org/abs/2210.02582> [6]

**Funding** *Neeldhara Misra:* The author is grateful for support from DST-SERB and IIT Gandhinagar. This work was partially supported by the ECR grant ECR/2018/002967.

*Prafullkumar Tale:* Part of the work was carried out when the author was a Post-Doctoral Researcher at CISA Helmholtz Center for Information Security, Germany, supported by the European Research Council (ERC) consolidator grant No. 725978 SYSTEMATICGRAPH.

**Acknowledgements** We are grateful for feedback from anonymous reviewers.



© Neeldhara Misra, Manas Mulpuri, Prafullkumar Tale, and Gaurav Viramgami; licensed under Creative Commons License CC-BY 4.0

42nd IARCS Annual Conference on Foundations of Software Technology and Theoretical Computer Science (FSTTCS 2022).

Editors: Anuj Dawar and Venkatesan Guruswami; Article No. 27; pp. 27:1–27:22



Leibniz International Proceedings in Informatics

Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

**1 Introduction**

The game of rendezvous with adversaries on a graph – RENDEZVOUS – is a natural dynamic version of the problem of finding a vertex cut between two vertices  $s$  and  $t$  introduced by Fomin, Golovach, and Thilikos [4]. The game is played on a finite undirected connected graph  $G$  by two players: *Facilitator* and *Divider*. Facilitator has two agents Romeo and Juliet that are initially placed in designated vertices  $s$  and  $t$  of  $G$ . Divider, on the other hand, has a team of  $k \geq 1$  agents  $D_1, \dots, D_k$  that are initially placed in some vertices of  $V(G) \setminus \{s, t\}$  chosen by him. We note that a single vertex can accommodate multiple agents of Divider.

Then the players make their moves by turn, starting with Facilitator. At every move, each player moves some of his/her agents to adjacent vertices or keeps them in their old positions. No agent can be moved to a vertex that is currently occupied by adversary’s agents. Both players have complete information about  $G$  and the positions of all the agents. Facilitator aims to ensure that Romeo and Juliet meet; that is, they are in the same vertex. The task of Divider is to prevent the rendezvous of Romeo and Juliet by maintaining  $D_1, \dots, D_k$  in positions that block the possibility to meet. Facilitator wins if Romeo and Juliet meet, and Divider wins if they succeed in preventing the meeting of Romeo and Juliet forever. This setup naturally leads to the following computational question.

**RENDEZVOUS**

**Input:** A graph  $G$  with two given vertices  $s$  and  $t$ , and a positive integer  $k$ .

**Question:** Can Facilitator win on  $G$  starting from  $s$  and  $t$  against Divider with  $k$  agents?

We will often refer to  $k$ , the number of agents employed by Divider to keep Romeo and Juliet separated, as the “solution size” for this problem.

**Known Results**

Fomin, Golovach, and Thilikos [4] initiated an extensive study of the computational complexity of RENDEZVOUS. They concluded that the problem is PSPACE-hard and co-W[2]-hard<sup>1</sup> when parameterized by the number of Divider’s agents, while also demonstrating an  $|V(G)|^{\mathcal{O}(k)}$  algorithm based on backtracking stages over the game arena. They also show that the problem admits polynomial time algorithms on chordal graphs and  $P_5$ -free graphs. A related problem considered is RENDEZVOUS IN TIME, which asks if Facilitator can force a win in at most  $\tau$  steps. It turns out that RENDEZVOUS IN TIME is co-NP-complete even for  $\tau = 2$  and is FPT when parameterized by  $\tau$  and the neighborhood diversity of the graph. The latter is an ILP-based approach and uses the fact that INTEGER LINEAR PROGRAMMING FEASIBILITY is FPT in the number of variables. We refer readers to [4], and references within, for more related problems.

The smallest number of agents that Divider needs to use to win on a graph  $G$  is called the “dynamic” separation number of  $G$ . We denote this by  $d_G(s, t)$ . Note that if  $s$  and  $t$  are adjacent or  $s = t$ , then  $d_G(s, t) := +\infty$ . The “static” separation number between  $s$  and

<sup>1</sup> We refer the reader to Appendix A.1 for the definitions of these complexity classes.

$t$ , the original positions of Facilitator’s agents, is simply the smallest size of a  $(s, t)$ -vertex cut, i.e, a subset of vertices whose removal disconnects  $s$  and  $t$ . We use  $\lambda_G(s, t)$  to denote the minimum size of a vertex  $(s, t)$ -separator in  $G$ . It is clear that  $d_G(s, t) \leq \lambda_G(s, t)$ , since positioning  $\lambda_G(s, t)$  many guards on the vertices of a  $(s, t)$ -vertex allows Divider to win the game right away. It turns out that  $d_G(s, t) = 1$  if and only if  $\lambda_G(s, t) = 1$ . However, there are examples of graphs where  $d_G(s, t)$  is arbitrarily smaller than  $\lambda_G(s, t)$  [4]. The results in [4] for chordal graphs and  $P_5$ -free graphs are based on the fact that in these graphs, it turns out that  $d_G(s, t) = \lambda_G(s, t)$ .

## Our Contributions

Given that the problem is hard in the solution size, often regarded the “standard” parameter, a natural approach is to turn to structural parameters of the input graph. One of the most popular structural parameters in the context of graphs is treewidth, which is a measure of how “tree-like” a graph is. XP and FPT algorithms parameterized by treewidth are natural generalizations of tractability on trees. Indeed, RENDEZVOUS is easy to solve on trees because  $\lambda_G(s, t) = 1$  for any distinct  $s$  and  $t$  when  $G$  is a tree and  $st \notin E(G)$ . The complexity of RENDEZVOUS parameterized by treewidth, however, is wide open – in particular it is not even known if the problem is in XP parameterized by treewidth.

Interestingly, it was pointed out in [4] that if the initial positions  $s$  and  $t$  are not in the same bag of a tree decomposition of width  $w$ , then the upper bound for the dynamic separation number by  $\lambda_G(s, t)$  together with the XP algorithm for the standard parameter can be employed to solve the problem in  $n^{O(w)}$  time. Thus, the question that was left open by Fomin, Golovach, and Thilikos was if the problem can be solved in the same time if  $s$  and  $t$  are in the same bag. Our first contribution is to answer this question in the negative by showing that RENDEZVOUS is in fact co-NP-hard even for graphs of *constant* treewidth. In fact, we show more:

► **Theorem 1.** RENDEZVOUS is co-NP-hard even when restricted to:

- graphs whose feedback vertex set number is at most 14, or
- graphs whose pathwidth is at most 16.

In particular, RENDEZVOUS is co-para-NP-hard parameterized by treewidth.

We obtain this hardness by a non-trivial reduction from the 3-DIMENSIONAL MATCHING problem. In the backdrop of this somewhat surprising result, we are motivated to pursue the question of the complexity of RENDEZVOUS for larger parameters. It turns out that even augmenting the feedback vertex set number or the pathwidth with the solution size is not enough. Specifically, we show that the problem is unlikely to admit an FPT-algorithm even when parameterized by these combined parameters.

► **Theorem 2.** RENDEZVOUS is co-W[1]-hard when parameterized by:

- the feedback vertex set number and the solution size, or
- the pathwidth and the solution size.

This result is shown by a parameter preserving reduction from the (MONOTONE) NAE-INTEGERS-3-SAT problem, which was shown to be W[1]-hard when parameterized by the number of variables by Bringmann et al. [1]. Note that with this, we have a reasonably complete understanding of RENDEZVOUS in the combined parameter. Indeed, recall that the problem is co-W[1]-hard and XP parameterized by the solution size alone, and co-para-NP-hard parameterized by the feedback vertex set number alone as shown above.

Given the above hardness, we consider RENDEZVOUS parameterized by the vertex cover number, a larger parameter compared to both the feedback vertex set number and pathwidth. The status of RENDEZVOUS with respect to the vertex cover parameterization was also left

open in [4]. We see that the problem admits a natural exponential kernel in this parameter when combined with the solution size, and is hence FPT in the combined parameter; however this kernel cannot be improved to a polynomial kernel under standard complexity-theoretic assumptions.

► **Theorem 3.** *RENDEZVOUS is FPT when parameterized by the vertex cover number of the input graph and the solution size. Moreover, the problem does not admit a polynomial kernel when parameterized by the vertex cover number and the solution size unless  $\text{NP} \subseteq \text{co-NP}/\text{poly}$ .*

We briefly describe the intuition for the exponential kernel with respect to the vertex cover number. Suppose the graph  $G$  has a vertex cover  $X$ , where  $|X| \leq \ell$ , and, one may assume, without loss of generality, that  $s, t \in X$ . Further, for any  $Y \subseteq X$ , let  $I_Y$  denote the set of all vertices in  $G \setminus X$  whose neighborhood in  $X$  is exactly  $Y$ . It is not hard to see that if  $|I_Y| > k$ , then one might as well “curtail” the set to  $k + 1$  vertices without changing the instance. This leads to an exponential kernel in the combined parameter. It is also true that  $k$  is bounded, without loss of generality, by  $\ell$  and the size of the common neighborhood of  $s$  and  $t$  to begin with; however, it is unclear if  $k$  can always be bounded by some function of the vertex cover alone. The kernelization lower bound follows from observing the structure of the reduced instance in the reduction used in [4] to prove that problem is  $\text{co-W}[2]$ -hard when parameterized by the solution size.

Finally, we present polynomial time algorithms on two restricted cases.

► **Theorem 4.** *RENDEZVOUS can be solved in polynomial time on the classes of treewidth at most two graphs and grids.*

Recall that the polynomial time algorithm on the classes of trees, chordal graphs, and  $P_5$ -free graphs is obtained by proving that the size of dynamic separator is same as that of separator. In case of grids, we present a winning strategy for Divider for any non-trivial instances. This makes grids unique graph class in which the problem admits polynomial time algorithm even when dynamic separator can be smaller than separator.

**Organization of the paper.** Due to lack of space, we defer several technical proofs, the polynomial time algorithms, and the preliminaries to the full version [6]. We describe the main intuitions for the proofs of Theorem 1 Section Section 2 We present a formal proof of Theorem 2 in Section 3. The proof of Theorem 3 partially discussed in Section 4. The hardness result can be found in [6].

## 2 co-para-NP-hardness Parameterized by FVS and Pathwidth

In this section, we prove that RENDEZVOUS is paraNP-hard when parameterized by the feedback vertex set number and the pathwidth of the input graph. To do that, we present a parameter preserving reduction from the 3-DIMENSIONAL MATCHING problem, which is known to be NP-hard [5, SP 1]. For notational convenience, we work with the following definition of the problem. An input consists of a universe  $\mathcal{U} = \{\alpha, \beta, \gamma\} \times [n]$ , a family  $\mathcal{F} = \{A_1, A_2, \dots, A_m\}$  of subsets of  $\mathcal{U}$  such that for every  $j \in [m]$ , set  $A_j = \{(\alpha, a_1), (\beta, b_1), (\gamma, c_1)\}$  for some  $a_1, b_1, c_1 \in [n]$ . The goal is to find a subset  $\mathcal{F}' \subseteq \mathcal{F}$  that covers  $\mathcal{U}$  (and contains exactly  $n$  sets).

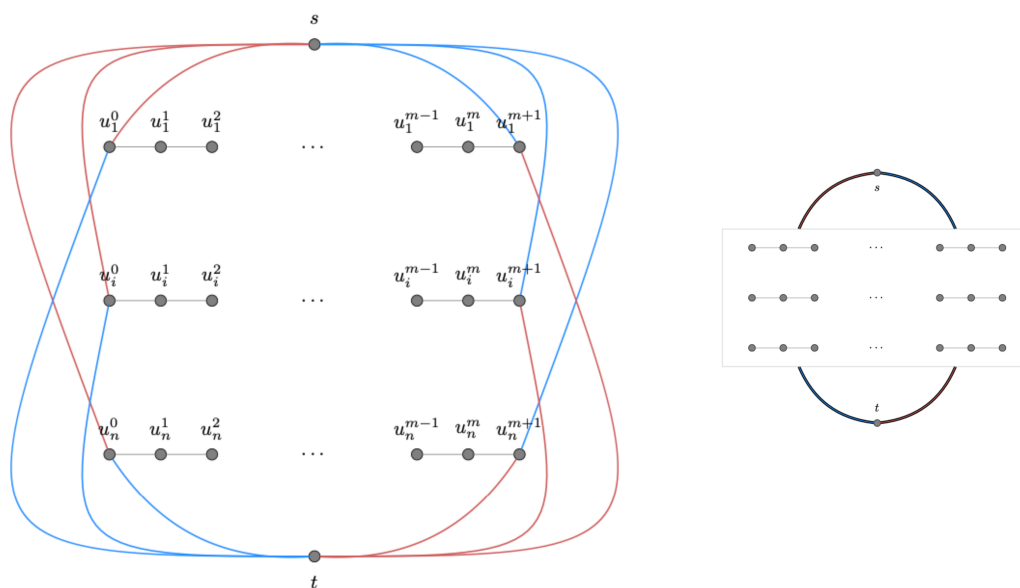
**Reduction**

The reduction takes as input an instance  $(\mathcal{U}, \mathcal{F})$  of 3-DIMENSIONAL MATCHING and returns an instance  $(G, s, t, k)$  of RENDEZVOUS. It defines  $M = n^2 + m^2$  where  $n = |\mathcal{U}|/3$  and  $m = |\mathcal{F}|$ . We construct the graph  $G$  as follows: (c.f. Figures 1–4).

**The Base Gadget.** It starts by adding special vertices  $s$  and  $t$  and two more vertices  $g_1$  and  $g_2$ , and makes them common neighbours of  $s$  and  $t$ . We use  $P[u, v, d]$  to denote a simple path from  $u$  to  $v$  that contains  $d$  many internal vertices.

- For every  $i \in [n]^2$ , it adds the following simple paths:
  - $P[u_i^0, u_i^{m+1}, m]$ ,
  - $P[s, u_i^0, m]$ ,  $P[s, u_i^{m+1}, m]$ ,  $P[t, u_i^0, m]$ , and  $P[t, u_i^{m+1}, m]$ .

See Figure 1 for an illustration.



■ **Figure 1** (Left) The base gadget except the guard vertices  $g_1$  and  $g_2$  (which are not shown for clarity). Each red and blue path has  $m$  internal vertices. (Right) Schematic representation.

**Encoding Elements.** The reduction constructs a symmetric graph to encode elements in  $\mathcal{U}$  and has “left-side” and “right-side.” It starts by adding vertices  $\{\alpha^\ell, \beta^\ell, \gamma^\ell\}$  and  $\{\alpha^r, \beta^r, \gamma^r\}$ .

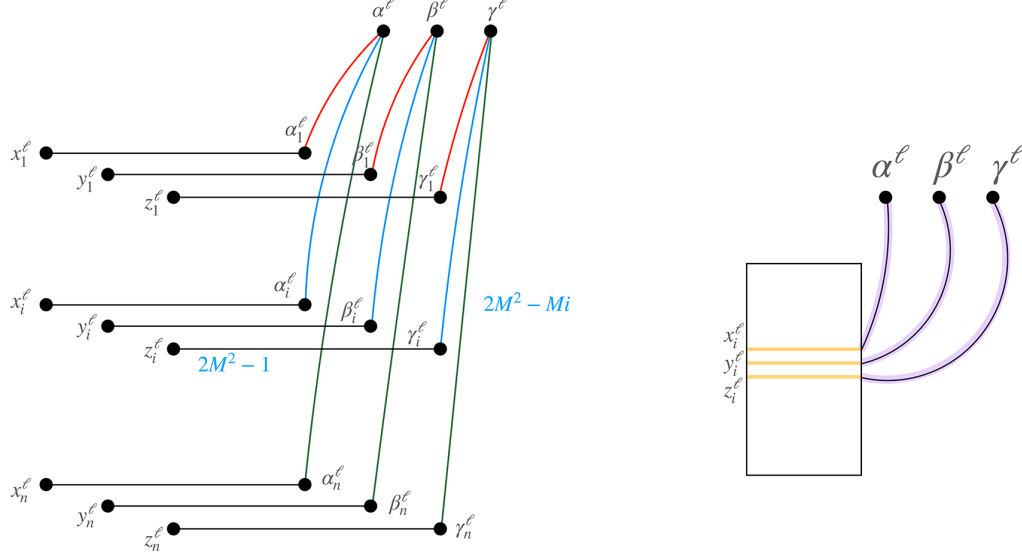
- For every  $i \in [n]$ , it adds six vertices in  $\{\alpha_i^\ell, \beta_i^\ell, \gamma_i^\ell\} \cup \{\alpha_i^r, \beta_i^r, \gamma_i^r\}$ , and the following simple paths:
  - $P[\alpha^\ell, \alpha_i^\ell, M^2 - M \cdot i]$ ,  $P[\beta^\ell, \beta_i^\ell, M^2 - M \cdot i]$ ,  $P[\gamma^\ell, \gamma_i^\ell, M^2 - M \cdot i]$ ,
  - $P[\alpha^r, \alpha_i^r, M^2 + M \cdot i]$ ,  $P[\beta^r, \beta_i^r, M^2 + M \cdot i]$ , and  $P[\gamma^r, \gamma_i^r, M^2 + M \cdot i]$ .

Note that the number of internal vertices in paths from  $\alpha^\ell$  to  $\alpha_i^\ell$  and from  $\alpha^r$  to  $\alpha_i^r$ , and similar such pairs, are different and depend on  $i$ .

<sup>2</sup> We use  $i$  as well as  $a_1, b_1, c_1$  as running variables in set  $[n]$ . We reserve later types of variables for the integer part of elements in sets  $\mathcal{F}$ .

## 27:6 Romeo and Juliet Meeting in Forest like Regions

- For every  $i \in [n]$ , it adds six vertices  $\{x_i^\ell, y_i^\ell, z_i^\ell\} \cup \{x_i^r, y_i^r, z_i^r\}$ , and the following simple paths:
    - $P[x_i^\ell, \alpha_i^\ell, 2M^2 - 1]$ ,  $P[y_i^\ell, \beta_i^\ell, 2M^2 - 1]$ ,  $P[z_i^\ell, \gamma_i^\ell, 2M^2 - 1]$ ,
    - $P[x_i^r, \alpha_i^r, 2M^2 - 1]$ ,  $P[y_i^r, \beta_i^r, 2M^2 - 1]$ , and  $P[z_i^r, \gamma_i^r, 2M^2 - 1]$ .
- See Figure 2 for an illustration.



■ **Figure 2** (Left) The left side of the gadget is added to encode elements in  $\mathcal{U}$ . The number of internal vertices in each red, blue, and green path depends on  $i$ . The number of internal vertices in each yellow shaded path is  $2M^2 - 1$ . (Right) Schematic representation of the gadget.

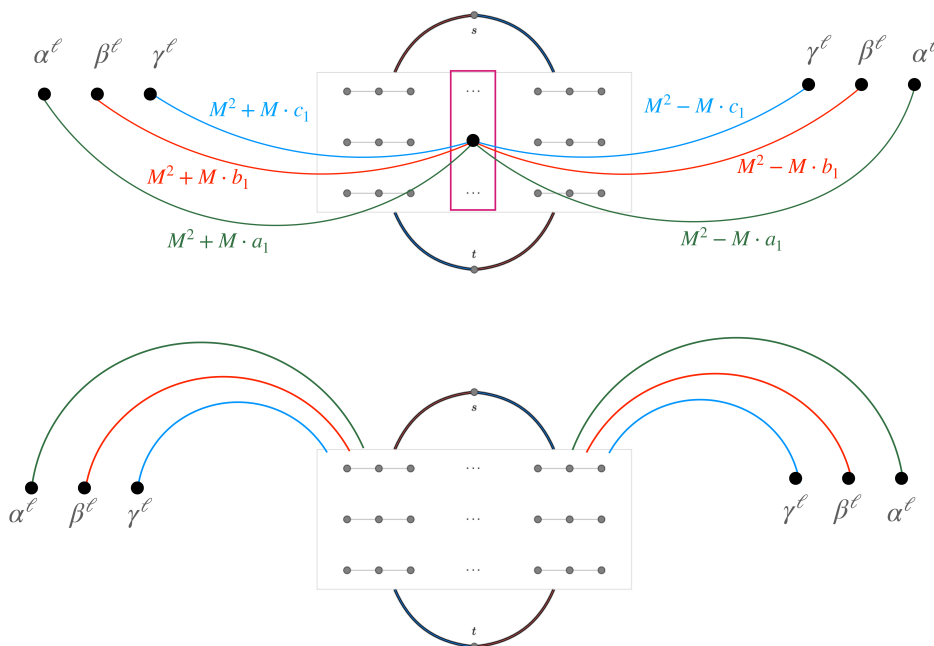
**Encoding sets.** The reduction adds simple paths to encode sets. Consider set  $A_j$  for some  $j \in [m]$ . Suppose the internal vertices of  $P[u_i^0, u_i^{m+1}, m]$  are denoted by  $u_i^j$  for every  $j \in [m]$ , and  $u_i^0$  is adjacent with  $u_i^1$  and  $u_i^{m+1}$  is adjacent with  $u_i^m$ . All the vertices in  $j^{\text{th}}$  'column' corresponds to set  $A_j$ . This, however, is not an encoding of set  $A_j$  as it does not provide any information about its elements. By the definition of the problem, set  $A_j$  has an element of the form  $(\alpha, a_1)$ . To encode this element, it adds  $2n$  many paths connecting  $j^{\text{th}}$  column to  $\alpha^\ell$  and to  $\alpha^r$ . The number of internal vertices in these paths depends on  $a_1$ . We encode the remaining two elements in  $A_j$  similarly. We formalise this construction as follows:

- For every  $j \in [m]$ , suppose  $A_j = \{(\alpha, a_1), (\beta, b_1), (\gamma, c_1)\}$ . Then, for every  $i \in [n]$ , the reduction adds the following six simple paths:
  - $P[\alpha^\ell, u_i^j, M^2 + M \cdot a_1]$ ,  $P[\beta^\ell, u_i^j, M^2 + M \cdot b_1]$ ,  $P[\gamma^\ell, u_i^j, M^2 + M \cdot c_1]$ ,
  - $P[\alpha^r, u_i^j, M^2 - M \cdot a_1]$ ,  $P[\beta^r, u_i^j, M^2 - M \cdot b_1]$ , and  $P[\gamma^r, u_i^j, M^2 - M \cdot c_1]$ .

See Figure 3 for an illustration.

**Critical vertices and connecting paths.** In the last phase of the reduction, it adds critical vertices and connect them to  $\{s, t\}$ , and also to  $x$ -type,  $y$ -type, and  $z$ -type ends of paths added while encoding elements in  $\mathcal{U}$ .

- For the special vertex  $s$ , it adds critical vertices, say  $s_\alpha^\ell, s_\beta^\ell$  and  $s_\gamma^\ell$ , on the left side.
  - It adds  $P[s, s_\alpha^\ell, 2M^2 + 1]$ ,  $P[s, s_\beta^\ell, 2M^2 + 1]$ , and  $P[s, s_\gamma^\ell, 2M^2 + 1]$ .
  - For every  $i \in [n]$ , it adds  $P[s_\alpha^\ell, x_i^\ell, 2M^2]$ ,  $P[s_\beta^\ell, y_i^\ell, 2M^2]$ , and  $P[s_\gamma^\ell, z_i^\ell, 2M^2]$ .



■ **Figure 3** (Top) Vertices added to encode sets in  $\mathcal{F}$ . The number of internal vertices in the paths depend on elements in  $A_j$  and are denoted next to it. (Bottom) Schematic representation of the gadget used in subsequent figures.

It adds the other critical vertices and paths symmetrically. We present them for the sake of completeness.

For the special vertex  $s$ , it adds critical vertices, say  $s_\alpha^r$ ,  $s_\beta^r$ , and  $s_\gamma^r$ , on the right side.

- It adds  $P[s, s_\alpha^r, 2M^2 + 1]$ ,  $P[s, s_\beta^r, 2M^2 + 1]$ , and  $P[s, s_\gamma^r, 2M^2 + 1]$ .
- For every  $i \in [n]$ , it adds  $P[s_\alpha^r, x_i^r, 2M^2]$ ,  $P[s_\beta^r, y_i^r, 2M^2]$ , and  $P[s_\gamma^r, z_i^r, 2M^2]$ .

For the special vertex  $t$ , it adds critical vertices, say  $t_\alpha^\ell$ ,  $t_\beta^\ell$ ,  $t_\gamma^\ell$ , on the left side.

- It adds  $P[t, t_\alpha^\ell, 2M^2 + 1]$ ,  $P[t, t_\beta^\ell, 2M^2 + 1]$ , and  $P[t, t_\gamma^\ell, 2M^2 + 1]$ .
- For every  $i \in [n]$ , it adds  $P[t_\alpha^\ell, x_i^\ell, 2M^2]$ ,  $P[t_\beta^\ell, y_i^\ell, 2M^2]$ ,  $P[t_\gamma^\ell, z_i^\ell, 2M^2]$ ,

For the special vertex  $t$ , it adds critical vertices, say  $t_\alpha^r$ ,  $t_\beta^r$ , and  $t_\gamma^r$ , on the right side.

- It adds  $P[t, t_\alpha^r, 2M^2 + 1]$ ,  $P[t, t_\beta^r, 2M^2 + 1]$ , and  $P[t, t_\gamma^r, 2M^2 + 1]$ .
- For every  $i \in [n]$ , it adds  $P[t_\alpha^r, x_i^r, 2M^2]$ ,  $P[t_\beta^r, y_i^r, 2M^2]$ , and  $P[t_\gamma^r, z_i^r, 2M^2]$ .

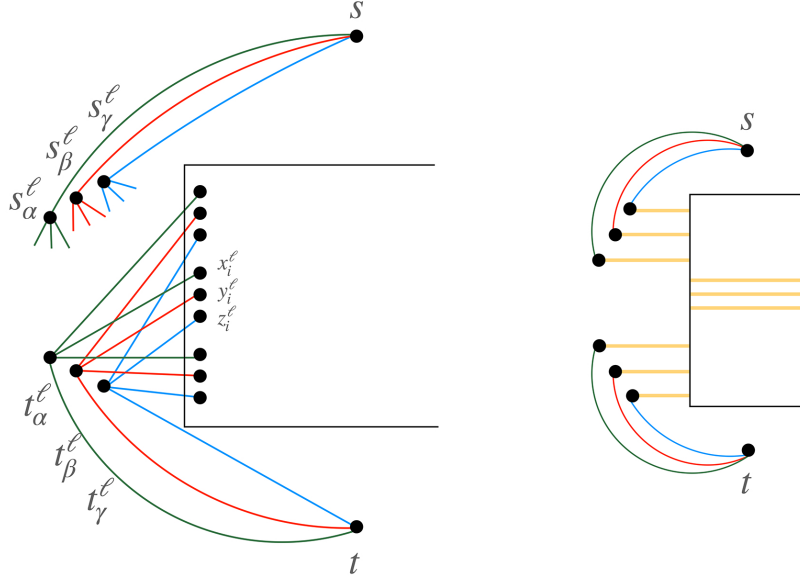
This completes the construction of the graph  $G$ . The reduction sets  $k = n + 2$  and returns  $(G, s, t, k)$  as the reduced instance of RENDEZVOUS.

### Intuition for the correctness

We present an intuition for the correctness of the reduction in the reverse direction. In other words, we state how the initial positions of the Divider’s agent correspond to sets and the elements they can cover. We start with determining the possible initial positions.

As  $g_1, g_2$  are common neighbors of  $s$  and  $t$ , Divider needs to put two agents on  $g_1$  and  $g_2$ . For the remaining  $n$  agents, consider the paths  $P[s, u_i^0, m]$  and  $P[u_i^0, t, m]$  or  $P[s, u_i^{m+1}, m]$  and  $P[u_i^{m+1}, t, m]$  for every  $i \in [n]$ . Facilitator can move both Romeo and Juliet to  $u_i^0$  or  $u_i^{m+1}$  in  $m$  steps. Hence, Divider needs to place remaining  $n$  agents at the positions that are at distance at most  $m$  simultaneously from  $u_i^0$  and  $u_i^{m+1}$ . We ensure that he needs to place





■ **Figure 4** (Left) Critical vertices added by the reduction. The number of internal vertices in the paths are fixed ( $2M^2 + 1$  or  $2M^2$ ). (Right) Schematic representation of the gadget.

an agent on an internal vertex of  $P[u_i^0, u_i^{m+1}, m]$  for every  $i \in [n]$ . This will correspond to selecting a set in  $\mathcal{F}$  in a solution. Formally, an agent at  $u_i^j$  for some  $j \in [m]$  corresponds to selecting  $A_j$  in the cover of  $\mathcal{U}$ . As there are  $n$  'rows', this will correspond to selecting  $n$  (different) sets from  $\mathcal{F}$ . Hence, the initial position of the Divider's agent will correspond to a collection of sets in  $\mathcal{F}$ .

Suppose for every  $i \in [n]$ , vertices in  $\{x_i^\ell, x_i^r\}$  correspond to element  $(\alpha, i) \in \mathcal{U}$ . Similarly, vertices in  $\{y_i^\ell, y_i^r\}$  correspond to  $(\beta, i)$ , and vertices in  $\{z_i^\ell, z_i^r\}$  correspond to  $(\gamma, i)$ . We say  $(\alpha, i)$  is covered if Divider can prevent Facilitator from moving both Romeo and Juliet at  $x_i^\ell$  as well as at  $x_i^r$ .

From the Facilitator's preservative, she has  $6n$  possible meeting points of the above form. She can make these choices in two phases. In the first phase, she can decide to move Romeo towards one of the six vertices in  $\{s_\alpha^\ell, s_\beta^\ell, s_\gamma^\ell\} \cup \{s_\alpha^r, s_\beta^r, s_\gamma^r\}$ . To win, she will have to move Juliet towards the corresponding vertices with respect to  $t$ . Suppose she moves Romeo towards  $s_\alpha^\ell$  and Juliet towards  $t_\alpha^\ell$ , i.e., Romeo along  $P[s, s_\alpha^\ell, 2M^2 + 1]$  and Juliet along  $P[t, t_\alpha^\ell, 2M^2 + 1]$ . She can move Romeo at  $s_\alpha^\ell$  and Juliet at  $t_\alpha^\ell$  in  $2M^2 + 2$  steps. At this point, she can make one of the  $n$  choices and decide to move both Romeo and Juliet towards  $x_i^\ell$  for some  $i \in [n]$ , i.e., Romeo along  $P[s_\alpha^\ell, x_i^\ell, 2M^2]$  and Juliet along  $P[t_\alpha^\ell, x_i^\ell, 2M^2]$  for some  $i \in [n]$ . See Figure 5 for relevant vertices.

From the Divider's perspective, he can see the first choice made by Facilitator. However, he has no information about her second choice until next  $2M + 2$  steps, i.e., until she moves Romeo at  $s_\alpha^\ell$  and Juliet at  $t_\alpha^\ell$ . Note that Facilitator can move Romeo from  $s_\alpha^\ell$  to  $x_i^\ell$  and Juliet from  $t_\alpha^\ell$  to  $x_i^\ell$  in  $2M^2 + 1$  steps. Divider can move an agent from  $\alpha_i^\ell$  to  $x_i^\ell$  in  $2M^2$  steps. Considering the initial positions of agents, he needs to ensure that one of its agents is present on  $\alpha_i^\ell$  for every  $i \in [n]$  in  $2M^2 + 2$  steps. For every  $i \in [n]$ , he needs to place an agent at  $u_i^j$  for some  $j \in [m]$  and  $i' \in [n]$  that he can move it to  $\alpha_i^\ell$  in  $2M^2 + 2$  steps. We remark that  $i$  may not be equal to  $i'$ .



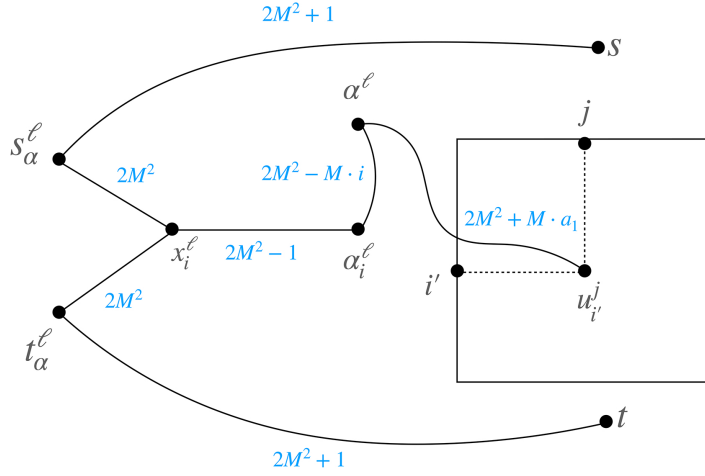


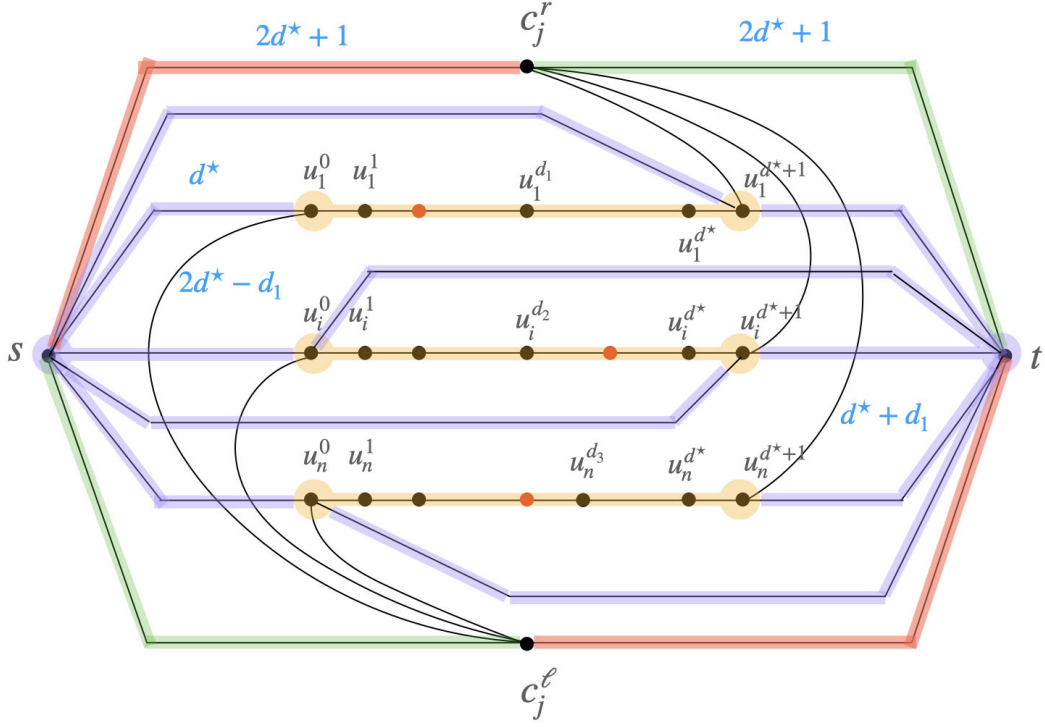
Figure 5 Vertices mentioned while presenting the intuition. The vertices near the paths indicate the number of internal vertices. Set  $A_j$  contains  $(\alpha, a_1)$ .

The only feasible way to do so is by moving the agent from  $u_{i'}^j$  to  $\alpha^\ell$  and then move it from  $\alpha^\ell$  to  $\alpha_i^\ell$ . Suppose  $(\alpha, a_1) \in A_j$  for some  $a_1 \in [n]$ . Recall that the number of internal vertices of path from  $u_{i'}^j$  to  $\alpha^\ell$  is  $M^2 + M \cdot a_1$  where as that of the path from  $\alpha^\ell$  to  $\alpha_i^\ell$  is  $M^2 - M \cdot i$ . Formally, the number of internal vertices in the path  $P[u_{i'}^j, \alpha^\ell, M^2 + M \cdot a_1] \circ P[\alpha^\ell, \alpha_i^\ell, M^2 - M \cdot i]$  is  $(M^2 + M \cdot a_1) + 1 + (M^2 - M \cdot i) = 2M^2 + 1 + M^2 \cdot (a_1 - i)$ . This implies that Divider can move an agent from  $u_{i'}^j$  to  $\alpha_i^\ell$  in  $2M^2 + 2 + M^2 \cdot (a_1 - i)$  steps. Hence, for every  $i \in [n]$ , Divider should place an agent at  $u_{i'}^j$  for some  $j \in [m]$  and  $i' \in [n]$  such that for  $(\alpha, a_1) \in A_j$ , we have  $a_1 \leq i$ . Using identical arguments and considering the number of internal vertices on the right side, we prove that for every  $i \in [n]$ , he needs to place an agent at  $u_{i''}^{j'}$  for some  $j' \in [m]$  and  $i'' \in [n]$  such that for  $(\alpha, a_2) \in A_{j'}$ , we have  $a_2 \geq i$ . Combining these two arguments, Divider needs to place an agent at  $u_{i'}^j$  such that for  $(\alpha, a_1) \in A_j$ , we have  $a_1 = i$ . Moving the agent from this position will prevent Facilitator from moving both Romeo and Juliet at  $x_i^\ell$  and  $x_i^r$ . This corresponds to selecting a set from  $\mathcal{F}$  to covers element  $(\alpha, i) \in \mathcal{U}$ . This concludes the intuition for the correctness of the reduction.

Consider set  $S := \{s, t\} \cup \{s_\alpha^\ell, s_\beta^\ell, s_\gamma^\ell\} \cup \{\alpha^\ell, \beta^\ell, \gamma^\ell\} \cup \{\alpha^r, \beta^r, \gamma^r\}$  in  $G$ . It is easy to verify that  $G - S$  is a forest, i.e., the feedback vertex set number of  $G$  is at most 14. Moreover, every connected component of  $G - S$  is either a path or a subdivided caterpillar. The paths correspond to the paths added while encoding elements in  $\mathcal{U}$  or while adding the critical paths. The subdivided caterpillars correspond to the base gadgets, and the path added while encoding sets in  $\mathcal{F}'$ . Note that the spine of the caterpillar is the path  $P[u_i^0, u_i^{m+1}, m]$  for some  $i \in [n]$  added as a part of base gadget. This implies that the pathwidth of the resulting graph is at most 16.

### 3 co-W[1]-hardness Parameterized by FVS, Pathwidth, and the Solution Size

In this section, we prove Theorem 2 that states RENDEZVOUS is co-W[1]-hard when parameterized by the feedback vertex set number or pathwidth and the solution size. To do that, we present a parameter preserving reduction from the (MONOTONE) NAE-INTEGGER-3-SAT



■ **Figure 6** The reduction adds a yellow shaded path for each variable. Each yellow, purple, or blue shaded path has  $d^*$  many internal vertices. The green and red shaded paths have  $2d^* + 1$  many internal vertices. The number of internal vertices in the remaining path in the figure depends on constants in the clause they are encoding. Note that vertices  $g_1, g_2$  and paths  $P[s, u_n^{d^*+1}, d^*], P[t, u_1^0, d^*]$  are not shown in the figure for clarity. The red vertices denote the positions of the agents.

problem. For notational convenience, we work with the following definition of the problem. An input consists of variables  $\mathcal{X} = \{x_1, \dots, x_n\}$  that each take a value in the domain  $\mathcal{D} = \{1, \dots, d^*\}$  and clauses  $\mathcal{C} = \{C_1, \dots, C_m\}$  of the form

$$\text{NAE}(x_{i_1} \leq d_1, x_{i_2} \leq d_2, x_{i_3} \leq d_3),$$

where  $d_1, d_2, d_3 \in [d^*]$ . Such a clause is satisfied if not all three inequalities are **True** and not all are **False** (i.e., they are “not all equal”). The goal is to find an assignment of the variables that satisfies all given clauses. Bringmann et. al. [1] proved that (MONOTONE) NAE-INTEGGER-3-SAT is  $W[1]$ -hard when parameterized by the number of variables.

### Reduction

The reduction takes as input an instance  $(\mathcal{X}, \mathcal{D}, \mathcal{C})$  of (MONOTONE) NAE-INTEGGER-3-SAT and returns an instance  $(G, s, t, k)$  of RENDEZVOUS. We construct the graph  $G$  as follows: (See Figure 6 for the overview of the constructed graph.)

**The Variable Gadget.** Recall that we use  $P[u, v, d]$  to denote a simple path from  $u$  to  $v$  that contains  $d$  many internal vertices. For every  $i \in [n]$ , it adds a simple path  $P[u_i^0, u_i^{d^*+1}, d^*]$ . Suppose the internal vertices of  $P[u_i^0, u_i^{d^*+1}, d^*]$  are denoted by  $u_i^d$  for every  $d \in [d^*]$ , and  $u_i^0$  is adjacent with  $u_i^1$  and  $u_i^{d^*+1}$  is adjacent with  $u_i^{d^*}$ .

**The Clause Gadget.** For every  $j \in [m]$ , the reduction adds two vertices  $c_j^\ell$  and  $c_j^r$ . Suppose  $C_j = \text{NAE}(x_{i_1} \leq d_1, x_{i_2} \leq d_2, x_{i_3} \leq d_3)$  for some  $j \in [m]$ . To encode the inequality  $x_{i_1} \leq d_1$ , the reduction adds simple paths  $P[c_j^\ell, u_{i_1}^0, 2d^* - d_1]$  and  $P[c_j^r, u_{i_1}^{d^*+1}, d^* + d_1]$ . It encodes the other two inequalities similarly. We highlight that the number of internal vertices in these simple paths depends on the constant in the inequalities they encode.

**Critical vertices and connecting paths.** The reduction adds special vertices  $s$  and  $t$  and two more vertices  $g_1$  and  $g_2$ , and makes them common neighbours of  $s$  and  $t$ .

- For every  $i \in [n]$ , it adds the following simple paths:
  - $P[s, u_i^0, d^*], P[s, u_i^{d^*+1}, d^*],$
  - $P[t, u_i^0, d^*], P[t, u_i^{d^*+1}, d^*].$
- For every  $j \in [m]$ , it adds the following simple paths:
  - $P[s, c_j^\ell, 2d^* + 1], P[s, c_j^r, 2d^* + 1],$
  - $P[t, c_j^\ell, 2d^* + 1], P[t, c_j^r, 2d^* + 1].$

This completes the construction of the graph  $G$ . The reduction sets  $k = n + 2$  and returns  $(G, s, t, k)$  as the reduced instance of RENDEZVOUS.

### Intuition for the correctness

We present an intuition for the correctness of the reduction. Recall that we use  $P[u, v, d_1] \circ P[v, w, d_2]$  to denote the unique path from  $u$  to  $w$  that contains  $v$ . Consider the paths  $P[s, u_i^0, d^*] \circ P[u_i^0, t, d^*]$  and  $P[s, u_i^{d^*+1}, d^*] \circ P[u_i^{d^*+1}, t, d^*]$  for every  $i \in [n]$  and paths  $P[s, c_j^\ell, 2d^* + 1] \circ P[c_j^\ell, t, 2d^* + 1]$  and  $P[s, c_j^r, 2d^* + 1] \circ P[c_j^r, t, 2d^* + 1]$  for every  $j \in [m]$ . As we will see, the only way Facilitator can win in Rendezvous Games with Adversaries is by moving Romeo and Juliet along with one of these  $2n + 2m$  paths. As  $g_1, g_2$  are common neighbors of  $s$  and  $t$ , Divider needs to put two of the  $k = n + 2$  agents on  $g_1$  and  $g_2$ . Suppose he puts the remaining  $n$  agents at some internal vertices of the paths added while encoding variables. He places the agents such that each path contains one of them. For example, the red vertices in Figure 6 corresponds to the positions of the agents on the paths added while encoding variables  $x_1, x_2$ , and  $x_3$ .

Suppose Divider places an agent at an internal vertex, say  $u_1^d$ , of  $P[u_1^0, u_1^{d^*+1}, d^*]$ . Facilitator can move Romeo and Juliet to either  $u_1^0$  or  $u_1^{d^*+1}$  in  $d^* + 1$  steps. The length of the path from  $u_1^0$  to  $u_1^d$  is  $d$  and the length of the path from  $u_1^{d^*+1}$  to  $u_1^d$  is  $d^* - d + 1$ .

- Divider can move the agent from  $u_1^d$  to  $u_1^0$  in at most  $d^*$  steps as  $d \leq d^*$ , and
- Divider can move the agent from  $u_1^d$  to  $u_1^{d^*+1}$  in at most  $d^*$  steps as  $d^* - d + 1 \leq d^*$ .

Recall that the simple path  $P[c_j^\ell, u_1^0, 2d^* - d_1]$ , as the notation suggests, has  $2d^* - d_1$  internal vertices. Hence, the path  $P[c_j^\ell, u_1^0, 2d^* - d_1] \circ P[u_1^0, u_1^d, d - 1]$  has  $(2d^* - d_1) + 1 + (d - 1)$  many internal vertices. Hence, the length of path from  $c_j^\ell$  to  $u_1^d$  is  $2d^* + 1 + d - d_1$ .

- Divider can move the agent from  $u_1^d$  to  $c_j^\ell$  in at most  $2d^* + 1$  steps only if  $d \leq d_1$ .

Consider symmetric arguments for  $c_j^r$ . The simple path  $P[c_j^r, u_1^{d^*+1}, d^* + d_1]$  has  $d^* + d_1$  many internal vertices. Hence, the path  $P[c_j^r, u_1^{d^*+1}, d^* + d_1] \circ P[u_1^{d^*+1}, u_1^d, d^* - d]$  has  $(d^* + d_1) + 1 + (d^* - d)$  many internal vertices. Hence, the length of the path from  $u_1^d$  to  $c_j^r$  is  $2d^* + 2 + d_1 - d$ .

- Divider can move the agent from  $u_1^d$  to  $c_j^r$  in at most  $2d^* + 1$  steps only if  $d > d_1$ .

Suppose there is a clause  $C_j \in \mathcal{C}$  such that  $C_j = \text{NAE}(x_1 \leq d_1, x_2 \leq d_2, x_3 \leq d_3)$ . Consider the two vertices  $c_j^\ell$  and  $c_j^r$  added while encoding  $C_j$ . Note that Facilitator can move Romeo and Juliet to either  $c_j^\ell$  or  $c_j^r$  in  $2d^* + 2$  steps. Moreover, apart from  $s$  and  $t$ , the only

branching points in paths  $P[s, c_j^\ell, 2d^* + 1] \circ P[c_j^\ell, t, 2d^* + 1]$  and  $P[s, c_j^r, 2d^* + 1] \circ P[c_j^r, t, 2d^* + 1]$  are  $c_j^\ell$  and  $c_j^r$ , respectively. Hence, Divider needs to place an agent that he can move to  $c_j^\ell$  in at most  $2d^* + 1$  steps. Similarly, he needs to place an agent that he can move to  $c_j^r$  in at most  $2d^* + 1$  steps. As we will see, Divider can only move the agents stationed at the paths corresponding to variables  $x_1, x_2$ , or  $x_3$  to  $c_j^\ell$  or  $c_j^r$  in at most  $2d^* + 1$  steps. Hence, he needs to place agents at the interior vertices, say  $u_1^{c_1}, u_2^{c_2}, u_3^{c_3}$ , of  $P[u_1^0, u_1^{d^*+1}, d^*]$ ,  $P[u_2^0, u_2^{d^*+1}, d^*]$  and  $P[u_3^0, u_3^{d^*+1}, d^*]$ , respectively, such that

- at least one of the inequalities in  $\{c_1 \leq d_1; c_2 \leq d_2; c_3 \leq d_3\}$  is **True**, and
- *simultaneously* at least one of the inequalities in  $\{c_1 > d_1; c_2 > d_2; c_3 > d_3\}$  is **True**.

This position of agents corresponds to the value of variables  $x_1, x_2, x_3$  in  $[d^*]$  that satisfy the clause  $C_j = \text{NAE}(x_1 \leq d_1, x_2 \leq d_2, x_3 \leq d_3)$ . In the following two lemmas, we formalize these intuitions.

► **Lemma 5.** *If  $(\mathcal{X}, \mathcal{D}, \mathcal{C})$  is a YES-instance of (MONOTONE) NAE-INTEGGER-3-SAT, then  $(G, s, t, n + 2)$  is a NO-instance of RENDEZVOUS.*

**Proof.** We show that if  $(\mathcal{X}, \mathcal{D}, \mathcal{C})$  is a YES-instance of (MONOTONE) NAE-INTEGGER-3-SAT, then Divider with  $n + 2$  agents can win in Rendezvous Game with Adversaries. Recall that  $n = |\mathcal{X}|$ , and  $m = |\mathcal{C}|$ . Suppose  $\psi : \mathcal{X} \rightarrow [d^*]$  be a satisfying assignment, and  $\psi(x_i) = d_i$  for every  $i \in [n]$ .

We describe a winning strategy for Dividers with the agents  $D_1, D_2, \dots, D_{n+2}$ . Initially, he puts  $D_i$  in the vertex  $u_i^{d_i}$ , for every  $i \in [n]$ , and  $D_{n+1}$  and  $D_{n+2}$  in  $g_1$  and  $g_2$  respectively. He does not move agents  $D_1, \dots, D_{n+2}$ , until Facilitator moves Romeo or Juliet from  $s$  or  $t$ , respectively. Suppose without loss of generality Facilitator first moves Romeo from  $s$  (she may or may not move Juliet from  $t$ ). By the construction, she can move Romeo either on the paths  $P[s, u_i^0, d^*]$ ,  $P[s, u_i^{d^*+1}, d^*]$  for some  $i \in [n]$  or on the paths  $P[s, c_j^\ell, 2d^* + 1]$ ,  $P[s, c_j^r, 2d^* + 1]$  for some  $j \in [m]$ .

Suppose Facilitator moves Romeo from  $s$  to a vertex on the path  $P[s, u_i^0, d^*]$  for some  $i \in [n]$ . Divider moves  $D_{n+1}$  from  $g_1$  to  $s$  and then towards  $u_i^0$  as she moves Romeo towards  $u_i^0$ . He also moves  $D_i$  to  $u_i^0$  in at most  $\psi(x_i)$  steps along the path  $P[u_i^0, u_i^{d^*+1}, d^*]$ . Facilitator needs at least  $d^* + 1$  steps to move both Romeo and Juliet in  $u_i^0$  starting from  $s$  and  $t$  respectively. As  $\psi(x_i) \leq d^*$ , Divider can move  $D_i$  to  $u_i^0$  before Facilitator can move both Romeo and Juliet to  $u_i^0$ . Hence, he can block Romeo by  $D_i$  and  $D_{n+1}$  on the path  $P[s, u_i^0, d^*]$ . Divider keeps moving  $D_i$  and  $D_{n+1}$  towards Romeo's position and in at most  $\psi(x_i) + d^* - 1$  steps Facilitator can not move Romeo. This implies Divider wins by keeping Romeo in its current position with its neighbors occupied by  $D_i$  and  $D_{n+1}$ . The argument also follows when Facilitator moves Romeo from  $s$  to a vertex on the path  $P[s, u_i^{d^*+1}, d^*]$  for some  $i \in [n]$  since Divider can move  $D_i$  to  $u_i^{d^*+1}$  in at most  $d^* - \psi(x_i) + 1 (\leq d^*)$  steps.

Suppose Facilitator moves Romeo from  $s$  to a vertex on the path  $P[s, c_j^\ell, 2d^* + 1]$  for some  $j \in [m]$ . Let  $C_j = \text{NAE}(x_{i_1} \leq d_1, x_{i_2} \leq d_2, x_{i_3} \leq d_3)$ . Since  $\psi$  is a satisfying assignment, it sets the values of variables such that at least one of the inequalities will be **True** and at least one of the inequalities will be **False**. We assume without loss of generality that  $\psi(x_{i_1}) \leq d_1$  and  $\psi(x_{i_2}) > d_2$ . Divider moves  $D_{i_1}$  to  $c_j^\ell$  in at most  $2d^* - d_1 + 1 + \psi(x_{i_1})$  steps through the path  $P[c_j^\ell, u_{i_1}^0, 2d^* - d_1] \circ P[u_{i_1}^0, u_{i_1}^{d^*+1}, d^*]$ . As in the previous case, he can move  $D_{n+1}$  from  $g_1$  to  $s$  and then keep moving towards  $c_j^\ell$  as Facilitator moves Romeo towards  $c_j^\ell$ . He can move  $D_{n+2}$  in a similar manner with respect to Juliet.

Facilitator can move both Romeo and Juliet to  $c_j^\ell$  in at least  $2d^* + 2$  steps starting from  $s$  and  $t$  respectively. Divider can move  $D_{i_1}$  to  $c_j^\ell$  before Romeo and Juliet as  $2d^* - d_1 + 1 + \psi(x_{i_1}) \leq 2d^* + 1$ . Hence, Romeo is blocked by  $D_{i_1}$  and  $D_{n+1}$  on the path  $P[s, c_j^\ell, 2d^* + 1]$

and Juliet cannot reach Romeo. Divider keeps moving  $D_{i_1}$  and  $D_{n+1}$  towards Romeo and in at most  $4d^* - d_1 + 1 + \psi(x_{i_1})$  steps Romeo cannot move. This implies Divider wins. The argument also follows when Facilitator moves Romeo from  $s$  to a vertex on the path  $P[s, c_j^r, 2d^* + 1]$  for some  $j \in [m]$  since Divider can move  $D_{i_2}$  to  $c_j^r$  in at most  $2d^* + 2 + d_2 - \psi(x_{i_2})$  ( $< 2d^* + 2$ ) steps. This implies that if  $(\mathcal{X}, \mathcal{D}, \mathcal{C})$  is a YES-instance of (MONOTONE) NAE-INTEGGER-3-SAT, then Divider with  $n + 2$  agents can win in Rendezvous Game with Adversaries, i.e.,  $(G, s, t, n + 2)$  is a NO-instance of RENDEZVOUS. ◀

► **Lemma 6.** *If  $(\mathcal{X}, \mathcal{D}, \mathcal{C})$  is a NO-instance of (MONOTONE) NAE-INTEGGER-3-SAT, then  $(G, s, t, n + 2)$  is a YES-instance of RENDEZVOUS.*

**Proof.** We show that if  $(\mathcal{X}, \mathcal{D}, \mathcal{C})$  is a NO-instance of (MONOTONE) NAE-INTEGGER-3-SAT, then Facilitator wins in at most  $2d^* + 2$  steps against Divider with  $n + 2$  agents.

We first consider two simple cases where Facilitator has an easy winning strategy. First, consider the case when Divider does not place his agents at  $g_1$  or  $g_2$ . Then, she can move Romeo and Juliet there and win in one step. Second, consider the case when there is  $i \in [n]$  such that none of Divider's agents is within distance  $d^*$  from  $u_i^0$  or from  $u_i^{d^*+1}$ . In the first sub-case, she can move Romeo and Juliet to  $u_i^0$  in  $d^* + 1$  steps through the paths  $P[s, u_i^0, d^*]$  and  $P[t, u_i^0, d^*]$ , respectively, and win. Similarly, in the second sub-case she can move Romeo and Juliet to  $u_i^{d^*+1}$  in  $d^* + 1$  steps through the paths  $P[s, u_i^{d^*+1}, d^*]$  and  $P[t, u_i^{d^*+1}, d^*]$ , respectively, and win.

In the remaining proof, we suppose that Divider places  $D_{n+1}$  at  $g_1$  and  $D_{n+2}$  at  $g_2$ . Moreover, for every  $i \in [n]$ , there is a Divider's agent within distance  $d^*$  from  $u_i^0$  and within distance  $d^*$  from  $u_i^{d^*+1}$ . Suppose from now that for every  $i \in [n]$ , there exists a Divider's agent within distance  $d^*$  from  $u_i^0$  and within distance  $d^*$  from  $u_i^{d^*+1}$ . By the construction and the fact that Divider can not place an agent at  $s$  or  $t$ , a single Divider's agent cannot be within distance  $d^*$  from both  $u_i^0$  and  $u_j^0$ , or  $u_i^{d^*+1}$  and  $u_j^{d^*+1}$ , or  $u_i^0$  and  $u_j^{d^*+1}$ , for  $i \neq j \in [n]$ . As Divider has  $n$  remaining agents, for every  $i \in [n]$ , there must be an agent, say  $D_i$ , within distance  $d^*$  from both  $u_i^0$  and  $u_i^{d^*+1}$ . This is possible only when for every  $i \in [n]$ ,  $D_i$  is on one of the internal vertices of the path  $P[u_i^0, u_i^{d^*+1}, d^*]$ . Suppose  $\phi : [n] \rightarrow [d^*]$  is the mapping corresponding to the initial position of the Divider's agents. Formally, for every  $i \in [n]$ , Divider places agent  $D_i$  on  $u_i^{\phi(i)}$ . For every  $i \in [n]$ , the initial position of  $D_i$  also represents a possible assignment of variable  $x_i$  in  $(\mathcal{X}, \mathcal{D}, \mathcal{C})$ .

We now define the Facilitator's strategy. Considering  $\mathcal{X} = \{x_1, \dots, x_n\}$  as the variables that each take a value in the domain  $\mathcal{D} = \{1, \dots, d^*\}$ , she constructs a collection  $\mathcal{C}$  of clauses such that for every  $j \in [m]$ , clause  $C_j = \text{NAE}(x_{i_1} \leq d_1, x_{i_2} \leq d_2, x_{i_3} \leq d_3)$ , where  $x_{i_1}, x_{i_2}, x_{i_3} \in \mathcal{X}$  for some  $d_1, d_2, d_3 \in [d^*]$ . Alternately, she reverse-engineers the process used by the reductions to encode clauses. She also constructs an assignment  $\psi : \mathcal{X} \rightarrow \mathcal{D} = [d^*]$  by considering the initial positions of agents  $D_1, D_2, \dots, D_n$ . Formally,  $\psi(x_i) = \phi(i)$  for every  $i \in [n]$ . It then determines whether the following statements are **True**.

1. For some clause  $C_j = \text{NAE}(x_{i_1} \leq d_1, x_{i_2} \leq d_2, x_{i_3} \leq d_3)$ , all of the inequalities in  $\{\psi(x_{i_1}) \leq d_1; \psi(x_{i_2}) \leq d_2; \psi(x_{i_3}) \leq d_3\}$  are **True**, where  $j \in [m]$ .
2. For some clause  $C_j = \text{NAE}(x_{i_1} \leq d_1, x_{i_2} \leq d_2, x_{i_3} \leq d_3)$ , all of the inequalities in  $\{\psi(x_{i_1}) \leq d_1; \psi(x_{i_2}) \leq d_2; \psi(x_{i_3}) \leq d_3\}$  are **False**, where  $j \in [m]$ .

Facilitator has to make a critical choice in the first step where she has to decide about moving Romeo towards  $c_1^l, \dots, c_m^l, c_1^r, \dots, c_m^r$ . This choice depends on which of the above statement is **True** and for which clause it is **True**. If the first statement is **True** for the clause  $C_j \in \mathcal{C}$ , then she moves Romeo and Juliet towards  $c_j^r$ . Similarly, if the second statement is **True** for the clause  $C_j \in \mathcal{C}$ , then she moves Romeo and Juliet towards  $c_j^l$ .

To argue that this is indeed a winning strategy for Facilitator, we first argue that for any initial positions of Divider's agents, at least one of the two statements above is **True**. Assume the above two statements are **False** for all  $j \in [m]$ , which implies in all the clauses  $C_j \in \mathcal{C}$ , not all three inequalities are **True** and not all are **False**. Hence, all the clauses are satisfied by the assignment  $\psi$ . This, however, contradicts the fact that  $(\mathcal{X}, \mathcal{D}, \mathcal{C})$  is a NO-instance. Hence, for any initial positions of Divider's agents, at least one of the two sentences is **True**.

This allows Facilitator to make her choice. It remains to argue that Romeo and Juliet can meet at the vertex  $c_j^\ell$  or  $c_j^r$  which Facilitator has chosen. Suppose, one of the statements is **True** for the clause  $C_j$ . For notational convenience, suppose  $C_j = \text{NAE}(x_1 \leq d_1, x_2 \leq d_2, x_3 \leq d_3)$ .

Suppose  $\psi(x_1) \leq d_1, \psi(x_2) \leq d_2, \psi(x_3) \leq d_3$  (i.e. First statement is **True**). Then, as mentioned in the Facilitator's strategy, her choice will be to move Romeo and Juliet towards  $c_j^r$ . For  $i \in \{1, 2, 3\}$ , Divider needs at least  $d^* - \psi(x_i) + 1 + d^* + d_i + 1 \geq 2d^* + 2$  steps to move  $D_i$  from  $u_i^{\psi(x_i)}$  to  $c_j^r$  via the shortest path  $P[u_i^{\psi(x_i)}, u_i^{d^*+1}, d^* - \psi(x_i)] \circ P[u_i^{d^*+1}, c_j^r, d^* + d_i]$ . Note that, by the construction, the Divider's agents that are at distance less than or equal to  $2d^* + 2$  from  $c_j^r$  are  $D_1, D_2$  and  $D_3$ , only. Facilitator can move Romeo and Juliet to  $c_j^r$  in  $2d^* + 2$  steps through the paths  $P[s, c_j^r, 2d^* + 1]$  and  $P[t, c_j^r, 2d^* + 1]$ , respectively. Since Facilitator takes the first turn, she can move Romeo and Juliet to  $c_j^r$  before Divider's agents. Hence, Facilitator wins in  $2d^* + 2$  steps.

Suppose  $\psi(x_1) > d_1, \psi(x_2) > d_2, \psi(x_3) > d_3$  (i.e. Second statement is **True**). Then, as mentioned in the Facilitator's strategy, her choice will be to move Romeo and Juliet towards  $c_j^\ell$ . For  $i \in \{1, 2, 3\}$ , Divider needs at least  $\psi(x_i) - 1 + 1 + 2d^* - d_i + 1 > 2d^* + 1$  steps to move  $D_i$  from  $u_i^{\psi(x_i)}$  to  $c_j^\ell$  via the shortest path  $P[u_i^{\psi(x_i)}, u_i^0, \psi(x_i) - 1] \circ P[u_i^0, c_j^\ell, 2d^* - d_i]$ . Once again, by the construction, the Divider's agents that are at distance less than or equal to  $2d^* + 2$  from  $c_j^\ell$  are  $D_1, D_2$  and  $D_3$ . Facilitator moves Romeo and Juliet to  $c_j^\ell$  in  $2d^* + 2$  steps through the paths  $P[s, c_j^\ell, 2d^* + 1]$  and  $P[t, c_j^\ell, 2d^* + 1]$  respectively. Since Facilitator takes the first turn, Romeo and Juliet is moved to  $c_j^\ell$  before Divider agents and Facilitator wins in  $2d^* + 2$  steps.

This implies that if  $(\mathcal{X}, \mathcal{D}, \mathcal{C})$  is a NO-instance of (MONOTONE) NAE-INTEGGER-3-SAT, then Facilitator wins in at most  $2d^* + 2$  steps against Divider with  $n + 2$  agents, i.e.,  $(G, s, t, n + 2)$  is a YES-instance of RENDEZVOUS. ◀

By the construction, the number of agents is upper bounded by the number of variables in (MONOTONE) NAE-INTEGGER-3-SAT plus two. Consider the set  $S := \bigcup_{i \in [n]} \{u_i^0, u_i^{d^*+1}\} \cup \{s, t\}$  of  $2n + 2$  vertices in  $G$ . It is easy to verify that  $G - S$  is a collection of paths (corresponding to variable gadgets) and subdivided stars (centered at the vertices added while encoding the clauses). It is easy to verify that the pathwidth of a subdivided star is at most two. Hence, the feedback vertex set number and the pathwidth of the resulting graph are bounded by the linear function in the number of variables. Lemma 5, Lemma 6 and the fact that the reduction can be completed in the polynomial time in the size of input imply Theorem 2 which we restate here.

- **Theorem 2.** RENDEZVOUS is co-W[1]-hard when parameterized by:
- the feedback vertex set number and the solution size, or
  - the pathwidth and the solution size.

#### 4 FPT Parameterized by the Vertex Cover Number and Solution Size

In this section we focus on Theorem 3. Throughout this section, we assume that a vertex cover  $X$  of size  $\text{vc}(G)$  is given as a part of the input. We describe the FPT result here and defer the non-existence of the polynomial kernel, which follows from observing the properties of a known reduction in [4], to the full version [6].



► **Theorem 3.** RENDEZVOUS is FPT when parameterized by the vertex cover number of the input graph and the solution size. Moreover, the problem does not admit a polynomial kernel when parameterized by the vertex cover number and the solution size unless  $\text{NP} \subseteq \text{co-NP}/\text{poly}$ .

► **Reduction Rule 4.1.** Consider an instance  $(G, X, s, t, k)$  of RENDEZVOUS. If  $s = t$ ,  $st \in E(G)$ ,  $|N(s) \cap N(t)| > k$ , then return a trivial YES-instance.

For the rest of this discussion, we will assume that any instance  $(G, s, t, k)$  of RENDEZVOUS under consideration does *not* satisfy the premise of Reduction Rule 4.1, i.e, we assume that we are not dealing with trivial YES instances. Also, since the vertices  $s$  and  $t$  can always be added to the vertex cover and this only increases the parameter by two, we assume for simplicity – and without loss of generality – that  $s, t \in X$ .

We now introduce some notation. For a subset  $Y \subseteq X$ , let  $I_Y \subseteq G \setminus X$  denote the set of vertices in  $G \setminus X$  whose neighborhood is exactly  $Y$ . Note that  $\{I_Y\}_{Y \subseteq X}$  is a partition of  $G \setminus X$  into at most  $2^{\text{vc}(G)}$  many parts. For a vertex  $v \in G \setminus X$ , we use  $\mathcal{E}_{G,X}(v)$  to denote the part that  $v$  belongs to, in other words,  $\mathcal{E}_{G,X}(v) = I_{N(v)}$ . We now apply the following reduction rule.

► **Reduction Rule 4.2.** Consider an instance  $(G, X, s, t, k)$  of RENDEZVOUS. Repeat the following for each  $v \in G \setminus X$ . If  $|\mathcal{E}_{G,X}(v)| > k + 1$ , then choose any subset of exactly  $k + 1$  vertices from  $\mathcal{E}_{G,X}(v)$  and delete rest of the vertices from  $\mathcal{E}_{G,X}(v)$ .

► **Lemma 7.** Reduction Rule 4.2 is safe.

**Proof.** Let  $(G, X, s, t, k)$  denote the input instance, and let  $v \in G \setminus X$  be arbitrary but fixed. Further, let  $(H, X, s, t, k)$  denote the instance obtained by applying Reduction Rule 4.2 with respect to  $v$ . If  $|\mathcal{E}(v)| \leq k + 1$  in  $G$  then  $G = H$  and there is nothing to prove. Otherwise, let  $Q_v \subseteq \mathcal{E}_{G,X}(v)$  denote the set of vertices deleted by the application of the reduction rule with respect to  $v$ . Note that  $H = G \setminus Q_v$ . Also observe that  $|\mathcal{E}_{H,X}(v)| = k + 1$ .

To begin with, suppose the Facilitator has a winning strategy in  $G$ . Observe that the Facilitator can employ the same strategy in  $H$  as well, except when the strategy involves moving to a vertex  $u \in Q_v$ . However, since  $|\mathcal{E}_{H,X}(v)| = k + 1$ , we have that there is at least one vertex  $w$  in  $H \setminus X$  that has the same neighborhood as  $u$  and is not occupied by an agent of the Divider, since the Divider has only  $k$  agents at their disposal. The strategy, at this point, would remain valid if we were to replace  $u$  with  $w$ . If the strategy involved using two distinct vertices from  $Q_v$  in the same step, then note that we can modify the strategy and have the Facilitator's agents meet immediately at the vertex  $w$ .

On the other hand, if the Facilitator had a winning strategy in  $H$ , then it is easy to check that the Facilitator can win in  $G$  by mimicing the strategy directly. Another way to see this is the following. Suppose that the Divider had a winning strategy in  $G$ . Then observe that in any step, without loss of generality, if the Divider's agents occupy some vertices of  $\mathcal{E}_{G,X}(v)$ , we can replace this configuration with all of these agents on a single vertex of  $\mathcal{E}_{G,X}(v)$  outside  $Q_v$ . Thus any winning strategy for the divider in  $G$  can be adapted to a valid winning strategy in  $H$ . This concludes the argument for the equivalence of the two instances. ◀

► **Lemma 8.** RENDEZVOUS is FPT when parameterized by the vertex cover number and the solution size.

**Proof.** Observe that repeated applications of Reduction Rule 4.2 ensures that  $|V(G)| = |X| + |G \setminus X| \leq \text{vc}(G) + 2^{\text{vc}(G)} \cdot (k + 1)$ . Thus we have an exponential kernel in  $\text{vc}(G)$ , and the claim follows. ◀



## 5 Conclusion

In this work, we studied the game of rendezvous with adversaries on a graph introduced by Fomin, Golovach, and Thilikos [4]. The game is a natural dynamic version of the problem of finding a vertex cut between two vertices  $s$  and  $t$ . Given that the problem is  $W[2]$ -hard when parameterized by the natural parameter, i.e. the solution size, we continued studying structural parameters of the input graph initiated by Fomin et al. [4]. We proved, to our surprise, that the problem is co-NP-hard even when restricted to graphs whose feedback vertex set number is at most 14, or pathwidth is at most 16. In particular, we proved RENDEZVOUS is co-para-NP-hard parameterized by treewidth, thereby answering an open question by Fomin et al. [4]. It turns out that even augmenting the feedback vertex set number or the pathwidth with the solution size is not enough. Specifically, we proved that RENDEZVOUS is co- $W[1]$ -hard when parameterized by the feedback vertex set number and the solution size, or the pathwidth and the solution size. Towards the positive side, we proved that the problem admits a natural exponential kernel when parameterized by the vertex cover number and the solution size, however this kernel cannot be improved to a polynomial kernel under standard complexity-theoretic assumptions. Finally, we presented polynomial time algorithms on two restricted cases and proved that RENDEZVOUS can be solved in polynomial time on the classes of treewidth at most two graphs and grids.

While we addressed the structural parameterized by arguably the most well studied parameters, it remains interesting to study the parameterized complexity by other structural parameters. Amongst these, we highlight the following question: *Is RENDEZVOUS  $W[1]$ -hard when parameterized by the vertex cover number (only)?* We tend to believe it is indeed the case. To the best of our knowledge, the problems that are  $W[1]$ -hard when parameterized by the vertex cover number, like LIST COLORING, WEIGHTED  $(k, r)$ -CENTER, etc., have additional input arguments like lists or weights. We believe that the dynamic nature of the RENDEZVOUS problem might make it an exception to the above known trend.

---

## References

- 1 Karl Bringmann, Danny Hermelin, Matthias Mnich, and Erik Jan van Leeuwen. Parameterized complexity dichotomy for steiner multicut. *J. Comput. Syst. Sci.*, 82(6):1020–1043, 2016. doi:10.1016/j.jcss.2016.03.003.
- 2 Marek Cygan, Fedor V. Fomin, Lukasz Kowalik, Daniel Lokshtanov, Dániel Marx, Marcin Pilipczuk, Michal Pilipczuk, and Saket Saurabh. *Parameterized Algorithms*. Springer, 2015. doi:10.1007/978-3-319-21275-3.
- 3 Reinhard Diestel. *Graph Theory, 4th Edition*, volume 173 of *Graduate texts in mathematics*. Springer, 2012.
- 4 Fedor V. Fomin, Petr A. Golovach, and Dimitrios M. Thilikos. Can romeo and juliet meet? or rendezvous games with adversaries on graphs. In Lukasz Kowalik, Michal Pilipczuk, and Pawel Rzazewski, editors, *Graph-Theoretic Concepts in Computer Science - 47th International Workshop, WG 2021, Warsaw, Poland, June 23-25, 2021, Revised Selected Papers*, volume 12911 of *Lecture Notes in Computer Science*, pages 308–320. Springer, 2021. doi:10.1007/978-3-030-86838-3\_24.
- 5 M. R. Garey and David S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W. H. Freeman, 1979.
- 6 Neeldhara Misra, Manas Mulpuri, Prafullkumar Tale, and Gaurav Viramgami. Romeo and juliet meeting in forest like regions, 2022. doi:10.48550/arXiv.2210.02582.

## A Appendix

### A.1 Preliminaries

For a positive integer  $q$ , we denote the set  $\{1, 2, \dots, q\}$  by  $[q]$ . We use  $\mathbb{N}$  to denote the collection of all non-negative integers.

#### Graph theory

We use standard graph-theoretic notation, and we refer the reader to [3] for any undefined notation. For an undirected graph  $G$ , sets  $V(G)$  and  $E(G)$  denote its set of vertices and edges, respectively. We denote an edge with two endpoints  $u, v$  as  $uv$ . Unless otherwise specified, we use  $n$  to denote the number of vertices in the input graph  $G$  of the problem under consideration. Two vertices  $u, v$  in  $V(G)$  are *adjacent* if there is an edge  $uv$  in  $G$ . The *open neighborhood* of a vertex  $v$ , denoted by  $N_G(v)$ , is the set of vertices adjacent to  $v$ . The *closed neighborhood* of a vertex  $v$ , denoted by  $N_G[v]$ , is the set  $N_G(v) \cup \{v\}$ . We say that a vertex  $u$  is a *pendant vertex* if  $|N_G(v)| = 1$ . The *degree* of a vertex  $v$ , denoted by  $\deg_G(v)$ , is equal to the number of vertices in the open neighbourhood of  $v$ , i.e.,  $\deg_G(v) = |N_G(v)|$ . We omit the subscript in the notation for neighborhood if the graph under consideration is clear.

For a subset  $S$  of  $V(G)$ , we define  $N[S] = \bigcup_{v \in S} N[v]$  and  $N(S) = N[S] \setminus S$ . For a subset  $F$  of edges, we denote by  $V(F)$  the collection of endpoints of edges in  $F$ . For a subset  $S$  of  $V(G)$  (resp. a subset  $F$  of  $E(G)$ ), we denote the graph obtained by deleting  $S$  (resp. deleting  $F$ ) from  $G$  by  $G - S$  (resp. by  $G - F$ ). We denote the subgraph of  $G$  induced on the set  $S$  by  $G[S]$ .

A graph is *connected* if there is a path between every pair of distinct vertices. A subset  $S \subseteq V(G)$  is said to be a *connected set* if  $G[S]$  is connected.

A *simple path*, denoted by  $P[u, v, d]$ , is a non-empty graph  $G$  of the form  $V(G) = \{u, x_1, \dots, x_d, v\}$ , and  $E(G) = \{ux_1, x_1x_2, \dots, x_{d-1}x_d, x_dv\}$ , where  $u, v$ , and all  $x_i$ 's are distinct. The vertices  $\{x_1, x_2, \dots, x_d\}$  are the *internal vertices* of  $P[u, v, d]$ , and the vertices  $\{x_i : \deg(x_i) > 2, i \in [d]\}$ , i.e., internal vertices whose *degree* is strictly greater than 2 are the *branching points* of  $P[u, v, d]$ . We use  $P[u, v, d_1] \circ P[v, w, d_2]$  to denote the unique simple path from  $u$  to  $w$  that contains  $v$  and has  $d_1 + d_2 + 1$  many internal vertices.

A set of vertices  $Y$  is said to be an *independent set* if no two vertices in  $Y$  are adjacent. For a graph  $G$ , a set  $X \subseteq V(G)$  is said to be a *vertex cover* if  $V(G) \setminus X$  is an independent set. A set of vertices  $Y$  is said to be a *clique* if any two vertices in  $Y$  are adjacent. A vertex cover  $X$  is a *minimum vertex cover* if for any other vertex cover  $Y$  of  $G$ , we have  $|X| \leq |Y|$ . We denote by  $\text{vc}(G)$  the size of a minimum vertex cover of a graph  $G$ . For a graph  $G$ , a set  $X \subseteq V(G)$  is said to be a *feedback vertex set* if  $V(G) \setminus X$  does not contain a cycle. We denote by  $\text{fvs}(G)$  the size of a minimum feedback vertex set of a graph  $G$ .

A *path decomposition* of a graph  $G$  is a sequence  $\mathcal{P} = (X_1, X_2, \dots, X_r)$  of *bags*, where  $X_i \subseteq V(G)$  for each  $i \in [r]$ , such that the following conditions hold:

- $\bigcup_{i=1}^r X_i = V(G)$ . In other words, every vertex of  $G$  is in at least one bag.
- For every  $uv \in E(G)$ , there exists  $\ell \in [r]$  such that the bag  $X_\ell$  contains both  $u$  and  $v$ .
- For every  $u \in V(G)$ , if  $u \in X_i \cap X_k$  for some  $i \leq k$ , then  $u \in X_j$  also for each  $j$  such that  $i \leq j \leq k$ . In other words, the indices of the bags containing  $u$  form an interval in  $[r]$ .

The *width* of a path decomposition  $(X_1, X_2, \dots, X_r)$  is  $\max_{1 \leq i \leq r} |X_i| - 1$ . The *pathwidth* of a graph  $G$ , denoted by  $\text{pw}(G)$ , is the minimum possible width of a path decomposition of  $G$ .

## 27:18 Romeo and Juliet Meeting in Forest like Regions

A *tree decomposition* of a graph  $G$  is a pair  $\mathcal{T} = (T, \{X_t\}_{t \in V(T)})$ , where  $T$  is a tree whose every node  $t$  is assigned a vertex subset  $X_t \subseteq V(G)$ , called a *bag*, such that the following conditions hold:

- $\bigcup_{t \in V(T)} X_t = V(G)$ . In other words, every vertex of  $G$  is in at least one bag.
- For every  $uv \in E(G)$ , there exists a node  $t$  of  $T$  such that bag  $X_t$  contains both  $u$  and  $v$ .
- For every  $u \in V(G)$ , the set  $T_u = \{t \in V(T) : u \in X_t\}$ , i.e., the set of nodes whose corresponding bags contains  $u$ , induces a connected subtree of  $T$ .

The *width* of a tree decomposition  $\mathcal{T} = (T, \{X_t\}_{t \in V(T)})$  is  $\max_{t \in V(T)} |X_t| - 1$ . The *treewidth* of a graph  $G$ , denoted by  $\text{tw}(G)$ , is the minimum possible width of a tree decomposition of  $G$ .

A  $M \times N$  *grid* is the graph  $G$  of the form  $V(G) = \{(i, j) : i \in [M], j \in [N]\}$ , and  $E(G) = \{(i, j)(i', j') : |i - i'| + |j - j'| = 1, i, i' \in [M], j, j' \in [N]\}$ .

Let  $X$  and  $Y$  be multisets of vertices of a graph  $G$  (i.e.,  $X$  and  $Y$  can contain several copies of the same vertex). We say that  $X$  and  $Y$  of the same size are *adjacent* if there is a bijective mapping  $\alpha : X \rightarrow Y$  such that for  $x \in X$ , either  $x = \alpha(x)$  or  $x$  and  $\alpha(x)$  are adjacent in  $G$ .

### Parameterized complexity

An instance of a parameterized problem  $\Pi$  consists of an input  $I$ , which is an input of the non-parameterized version of the problem, and an integer  $k$ , which is called the *parameter*. A problem  $\Pi$  is said to be *fixed-parameter tractable*, or **FPT**, if given an instance  $(I, k)$  of  $\Pi$ , we can decide whether  $(I, k)$  is a YES-instance of  $\Pi$  in time  $f(k) \cdot |I|^{\mathcal{O}(1)}$ . Here,  $f : \mathbb{N} \mapsto \mathbb{N}$  is some computable function depending only on  $k$ . Parameterized complexity theory provides tools to rule out the existence of **FPT** algorithms under plausible complexity-theoretic assumptions. For this, a hierarchy of parameterized complexity classes  $\text{FPT} \subseteq \text{W}[1] \subseteq \text{W}[2] \cdots \subseteq \text{XP}$  was introduced, and it was conjectured that the inclusions are proper. The most common way to show that it is unlikely that a parameterized problem admits an **FPT** algorithm is to show that it is **W[1]** or **W[2]**-hard. It is possible to use reductions analogous to the polynomial-time reductions employed in classical complexity. Here, the concept of **W[1]**-hardness replaces the one of **NP**-hardness, and we need not only to construct an equivalent instance **FPT** time, but also to ensure that the size of the parameter in the new instance depends only on the size of the parameter in the original instance. These types of reductions are called *parameter preserving reductions*. For a detailed introduction to parameterized complexity and related terminologies, we refer the reader to the recent book by Cygan et al. [2].

A *reduction rule* is a polynomial-time algorithm that takes as input an instance of a problem and outputs another, usually reduced, instance. A reduction rule said to be *applicable* on an instance if the output instance and input instance are different. A reduction rule is *safe* if the input instance is a YES-instance if and only if the output instance is a YES-instance.

A *kernelization* of a parameterized problem  $\Pi_1$  is a polynomial algorithm that maps each instance  $(I_1, k_1)$  of  $\Pi_1$  to an instance  $I$  of  $\Pi_2$  such that (1)  $(I, k)$  is a YES-instance of  $\Pi_1$  if and only if  $I_2$  is a YES-instance of  $\Pi_2$ , and (2) the size of  $I_2$  is bounded by  $g(k)$  for a computable function  $g(\cdot)$ . We say a compression is a *polynomial compression* if  $g(\cdot)$  is a polynomial function, then we call it a *polynomial kernel*. It is known that a problem is **FPT** if and only if it admits a kernel (See, for example, [2, Lemma 2.2]).

## B Some Polynomial Cases

In this section, we focus on Theorem 4.

► **Theorem 4.** RENDEZVOUS can be solved in polynomial time on the classes of treewidth at most two graphs and grids.

We focus on graphs of treewidth at most two. In this case, we show that  $d_G(s, t) = \lambda_G(s, t)$ , which leads to RENDEZVOUS being polynomially solvable on this class of graphs based on standard algorithms for computing  $\lambda_G(s, t)$ .

► **Proposition 9.** If  $G$  is a connected graph of tree-width at most 2, then for every  $s, t \in V(G)$ ,  $d_G(s, t) = \lambda_G(s, t)$ .

**Proof.** Consider an instance  $(G, s, t, k)$  of RENDEZVOUS where  $G$  is a graph of treewidth at most 2. We recall that these are the series-parallel graphs, which are graphs with two distinguished vertices called terminals, formed recursively by two simple composition operations. Specifically, we have the following definitions. A two-terminal graph (TTG) is a graph with two distinguished vertices,  $s$  and  $t$  called source and sink, respectively. The parallel composition  $P_c = P_c(X, Y)$  of two TTGs  $X$  and  $Y$  is a TTG created from the disjoint union of graphs  $X$  and  $Y$  by merging the sources of  $X$  and  $Y$  to create the source of  $P_c$  and merging the sinks of  $X$  and  $Y$  to create the sink of  $P_c$ . The series composition  $S_c = S_c(X, Y)$  of two TTGs  $X$  and  $Y$  is a TTG created from the disjoint union of graphs  $X$  and  $Y$  by merging the sink of  $X$  with the source of  $Y$ . The source of  $X$  becomes the source of  $S_c$  and the sink of  $Y$  becomes the sink of  $S_c$ . A two-terminal series-parallel graph (TTSPG) is a graph that may be constructed by a sequence of series and parallel compositions starting from a set of copies of a single-edge graph  $K_2$  with assigned terminals. Finally, a graph is called series-parallel (SP-graph), if it is a TTSPG when some two of its vertices are regarded as source and sink.

The proof will proceed by induction on the number of vertices. We will do a case analysis for sequence of compositions used to arrive at the final graph  $G$ . In the base case, there is nothing to prove since  $G$  is simply an edge. We use  $x$  and  $y$  to denote the source and sink terminals, respectively. For the induction hypothesis, we assume that the claim is true for all series-parallel graphs with less than  $k$  vertices, where  $k \geq 2$ . Now, let  $G$  be a series-parallel graph having  $k$  vertices.

Suppose  $G$  is obtained by a series or parallel composition of graphs  $G_1$  and  $G_2$  and let  $x$  and  $y$  denote the source and sink terminals of  $G$ , while  $x_b$  and  $y_b$  denote the source and sink terminals of  $G_b$  for  $b \in \{1, 2\}$ . Note that  $G_1$  and  $G_2$  are series-parallel graphs having less than  $k$  vertices.

**Case 1:**  $s \in G_1$  and  $t \in G_2$ ;  $s \neq x_1, s \neq y_1; t \neq x_2, t \neq y_2$

**Case 1A: The composition is series.** In this case static and dynamic separation number is one:  $\{x\}$  or  $\{y\}$  (the joined terminal).

**Case 1B: The composition is parallel.** Consider the path  $s \rightarrow x \rightarrow t \rightarrow y \rightarrow s$ . Since  $s \neq t, s \neq x, s \neq y, t \neq x, t \neq y$ , the considered path is a closed walk containing  $s$  and  $t$  which forms a cycle. So,  $s$  and  $t$  lies on a cycle. So, the lower bound on the dynamic separator is 2. And the upper bound on the dynamic separator is also 2 as the static separator in this case is 2. So in this case static and dynamic separation number is two and is given by both terminals together:  $\{x, y\}$ .

**Case 2:**  $s \in G_1$  and  $t \in G_1$   $s \neq x_1, s \neq y_1; t \neq x_1, t \neq y_1$

**Case 2A: The composition is series.** In this case, suppose  $y_1$  and  $x_2$  are identified as  $g_{1,2}$ ; and  $x = x_1$  and  $y = y_2$ .

## 27:20 Romeo and Juliet Meeting in Forest like Regions

▷ Claim 10. Static  $s, t$  separators in  $G_1$  will also work in  $G$  and vice versa.

*Proof. Forward Direction.* Suppose  $G_1$  has a static  $(s, t)$  separator  $S_1$  of size  $k_1$ . So, there does not exist any path from  $s$  to  $t$  in  $G_1$  which does not contain any vertex of  $S_1$ . Now, for the supergraph  $G$ , all the paths between  $s$  and  $t$  that does not pass through  $G_2$  are already blocked by the static separator  $S_1$ . Further, the paths that pass through  $G_2$  will pass through the terminal vertex twice. So these paths will be  $s \rightarrow g_{1,2} \rightarrow$  some vertices of  $G_2 \rightarrow g_{1,2} \rightarrow t$ . Suppose these paths are not blocked by  $S_1$ , then there also exist a path  $s \rightarrow g_{1,2} \rightarrow t$  in  $G_1$  that are not blocked by  $S_1$ , which contradicts the assumption that  $S_1$  is a static  $(s, t)$  separator in  $G_1$ . So, these paths are also blocked by  $S_1$ , which implies that  $S_1$  is also the static separator of  $G$ .

*Backward Direction.* Suppose  $G$  has a static  $(s, t)$  separator  $S_1$  of size  $k_1$ . Taking the vertices from  $G_2$  in the static separator can only block paths of the type  $s \rightarrow g_{1,2} \rightarrow$  some vertices of  $G_2 \rightarrow g_{1,2} \rightarrow t$ . So,  $S_1$  can not contains more than one vertex from the graph  $G_2$ , else those vertices can be replaced by  $g_{1,2}$  which will result in a smaller sized static  $(s, t)$  separator of  $G$ . Hence,  $S_1$  can contain at max one vertex from  $G_2$ . Observe that if  $S_1$  does not contain any vertex of  $G_2$ , then the same  $S_1$  is also a static  $(s, t)$  separator in  $G_1$ . If  $S_1$  contains exactly one vertex from  $G_2$ , then that vertex can be replaced by  $g_{1,2}$  to form a same sized static  $(s, t)$  separator in  $G_1$ . ◁

▷ Claim 11.  $d_G(s, t) = \lambda_G(s, t) = k_1$ .

*Proof.* Suppose the claim is not true. Then we need fewer than  $k_1$  guards in  $G$ , say  $k_1 - 1$  guards are enough to separate  $s$  from  $t$  in  $G$ . But then this will also be a valid strategy in  $G_1$ , contradicting the induction hypothesis from which we know that  $d_{G_1}(s, t) = \lambda_{G_1}(s, t) = k_1$ . ◁

**Case 2B: The composition is parallel.** In this case, we have that  $y_1$  and  $y_2$  join into  $y$  and  $x_1$  and  $x_2$  join into  $x$ .

▷ Claim 12.  $\lambda_{G_1}(s, t) \leq \lambda_G(s, t) \leq \lambda_{G_1}(s, t) + 1$ .

*Proof.* The first inequality follows from the fact that  $G$  is a supergraph of  $G_1$ . For the second inequality, note that a separation of  $s$  and  $t$  in  $G$  can be achieved by adding one of the terminal vertices to the static separator in  $G_1$ . ◁

▷ Claim 13.  $k_1 \leq d_G(s, t) = \lambda_G(s, t) \leq k_1 + 1$ .

*Proof.* Suppose the claim is not true. Suppose  $\lambda_G(s, t) = k_1$ , and we need fewer than  $k_1$  guards in  $G$ , say  $k_1 - 1$  guards are enough to separate  $s$  from  $t$  in  $G$ . But then this will also be a valid strategy in  $G_1$ , contradicting the induction hypothesis from which we know that  $d_{G_1}(s, t) = \lambda_{G_1}(s, t) = k_1$ . Suppose  $\lambda_G(s, t) = k_1 + 1$ , and we need fewer than  $k_1 + 1$  guards in  $G$ , say  $k_1$  guards are enough to separate  $s$  from  $t$  in  $G$ . If  $\lambda_G(s, t)$  has more than one vertex of  $G_2$ , then it will contradict the induction hypothesis from which we know that  $\lambda_G(s, t) = k_1$ . If it does not have any vertex of  $G_2$  then we can replace this separator with the union of static  $(s, t)$  separator of  $G_1$

and the vertex  $x$ . And if it contains exactly one vertex of  $G_2$ , then that vertex can be replaced by one of the terminals  $x$  or  $y$  and it will still be a valid static  $(s, t)$  separator in  $G$ . Observe that, to separate  $s$  and  $t$  in  $G$ , at least one guard must be on one of the paths of type  $s \rightarrow x \rightarrow$  some vertices of  $G_2 \rightarrow y \rightarrow t$ . If not then there is no guard which can block  $x, y$ , and the vertices of  $G_2$  and so it will not be a valid separator. Additionally, this vertex is not needed in the dynamic  $(s, t)$  separator of  $G_1$ . Hence we can obtain a  $k_1 - 1$  size dynamic  $(s, t)$  separator in  $G_1$  by eliminating this vertex, contradicting the induction hypothesis.  $\triangleleft$

**Case 3:  $s$  is one of the terminals and  $t \in G_1$  or  $t \in G_2$ ;  $s = x$  or  $s = y$ ;  $t \neq x_1, t \neq y_1$**

**Case 3A: The composition is series.**

1. If  $s = x$  and  $t \in G_2$  or if  $s = y$  and  $t \in G_1$ , then the vertex at the junction of the composition is a separator of size one.
2. If  $s = x$  and  $t \in G_1$  or  $s = y$  and  $t \in G_2$ , then the static separator of  $s$  and  $t$  in  $G$  is the static separator of  $s$  and  $t$  in  $G_1$  or the static separator of  $s$  and  $t$  in  $G_2$ , of size  $k_1$  or  $k_2$  respectively.

$\triangleright$  **Claim 14.** If  $s = x$  and  $t \in G_1$ , then  $d_G(s, t) = \lambda_G(s, t) = k_1$ , while if  $s = y$  and  $t \in G_2$ , then  $d_G(s, t) = \lambda_G(s, t) = k_2$ .

*Proof.* Consider the first statement and suppose the claim is not true. Then we need fewer than  $k_1$  guards in  $G$ , say  $k_1 - 1$  guards are enough to separate  $s$  from  $t$  in  $G$ . But then this will also be a valid strategy in  $G_1$ , contradicting the induction hypothesis. The same argument works for the second statement as well.  $\triangleleft$

**Case 3B: The composition is parallel.**

$\triangleright$  **Claim 15.**  $\lambda_G(s, t) \leq \lambda_{G_1}(s, t) + 1$ , if  $t \in G_1$  and  $\lambda_G(s, t) \leq \lambda_{G_2}(s, t) + 1$ , if  $t \in G_2$ . This argument in this case is analogous to Case 2B.

**Case 4.  $s$  and  $t$  are both terminals.**

**Case 4A: The composition is series.**

1. If  $s = x; t = y$ , then the vertex at the junction of the composition is a static separator of size one.
2. If  $s = x; t = z$ , where  $z$  denotes the vertex at the junction of the composition, then a static separator of  $s$  and  $t$  in  $G_1$  is also a static separator of  $s$  and  $t$  in  $G$ .
3. If  $s = z; t = y$ , where  $z$  is the same as before, then a static separator of  $s$  and  $t$  in  $G_2$  is also a static separator of  $s$  and  $t$  in  $G$ .

$\triangleright$  **Claim 16.** If  $s = x$  and  $t = z$ , then  $d_G(s, t) = \lambda_G(s, t) = k_1$ , while if  $s = z$  and  $t = y$ , then  $d_G(s, t) = \lambda_G(s, t) = k_2$ .

*Proof.* Consider the first statement and suppose the claim is not true. Then we need fewer than  $k_1$  guards in  $G$ , say  $k_1 - 1$  guards are enough to separate  $s$  from  $t$  in  $G$ . But then this will also be a valid strategy in  $G_1$ , contradicting the induction hypothesis. The same argument works for the second statement as well.  $\triangleleft$

## 27:22 Romeo and Juliet Meeting in Forest like Regions

**Case 4B: The composition is parallel, i.e.  $s = x; t = y$ .** In this case, note that the size of the static separator of  $s$  and  $t$  in  $G = \lambda_{G_1}(s, t) + \lambda_{G_2}(s, t)$ . Indeed, any smaller subset would have either fewer than  $\lambda_{G_1}(s, t)$  vertices in  $G[V(G_1)]$  or fewer than  $\lambda_{G_2}(s, t)$  vertices in  $G[V(G_2)]$ , implying the existence of an unblocked path from  $s$  to  $t$  via  $G_1$  or  $G_2$

▷ Claim 17.  $d_G(s, t) = \lambda_G(s, t) = k_1 + k_2$ .

Proof. Suppose the claim is not true. Then we need fewer than  $k_1 + k_2$  guards in  $G$ , say  $k_1 + k_2 - 1$  guards are enough to separate  $s$  from  $t$  in  $G$ . But then this will also be a valid strategy in  $G_1$  (or  $G_2$ ) with  $< k_1$  (or  $< k_2$ ) guards, contradicting the induction hypothesis. ◁

So, for all the cases  $\lambda_G(s, t) = d_G(s, t)$ , and this concludes our argument. ◀