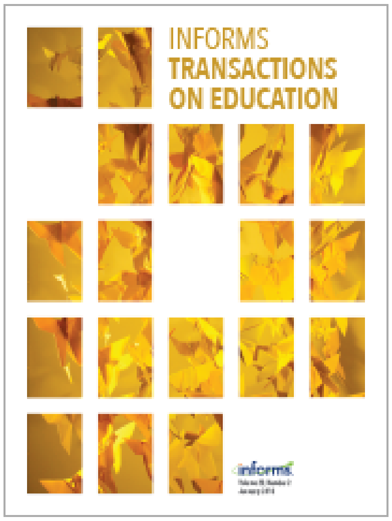


INFORMS Transactions on Education

Publication details, including instructions for authors and subscription information:
http://pubsonline.informs.org

Odd-Length Exchanges in ABO-Only Kidney Exchange: A Feasibility Puzzle for the Classroom

Gabriel Teodoro, Hamidreza Validi

informs
https://pubsonline.informs.org/journal/ited

INFORMS TRANSACTIONS ON EDUCATION
Articles in Advance, pp. 1–6
ISSN 1532-0545 (online)

Odd-Length Exchanges in ABO-Only Kidney Exchange: A Feasibility Puzzle for the Classroom

Gabriel Teodoro,[a] Hamidreza Validi[a,*]

[a] Department of Industrial, Manufacturing & Systems Engineering, Texas Tech University, Lubbock, Texas 79409
*Corresponding author
Contact: gteodoro@ttu.edu (GT); hvalidi@ttu.edu, https://orcid.org/0000-0002-7983-7262 (HV)

Received: November 7, 2025
Revised: January 23, 2026; March 1, 2026
Accepted: April 19, 2026
Published Online in Articles in Advance: July 6, 2026

https://doi.org/10.1287/ited.2025.0193

Abstract. Most classroom puzzles in operations research emphasize optimizing an objective; here, we instead pose a *feasibility* puzzle rooted in kidney exchange programs. We ask whether odd-length cycles—three-way exchanges as the canonical case—can arise when every donor-recipient pair is internally incompatible and compatibility is defined by ABO blood type alone. We formalize the puzzle as a mixed-integer feasibility model that encodes assignment, incompatibility, and donor-to-next-recipient implication constraints. Simple variable fixings collapse the model to a reduced formulation whose structure reveals a bipartition of the cycle-capable nodes (A to B and B to A), thereby precluding any odd cycle and certifying infeasibility for the three-way case; the same logic extends to all odd lengths ≥ 5. We provide a concise Python and Gurobi implementation and an undergraduate classroom activity that uses the model to contrast feasibility reasoning with optimization thinking, connect blood-compatibility rules to mathematical constraints, and practice modular indexing on cyclic structures. The puzzle thus serves both as a correctness certificate for the ABO-only setting and as a compact teaching vehicle linking graph structure and integer programming.

Supplemental Material: Data is available at https://www.informs.org/Publications/Subscribe/Access-Restricted-Materials.

Keywords: kidney exchange program • blood-type compatibility • puzzle-based learning • feasibility modeling • mixed-integer programming

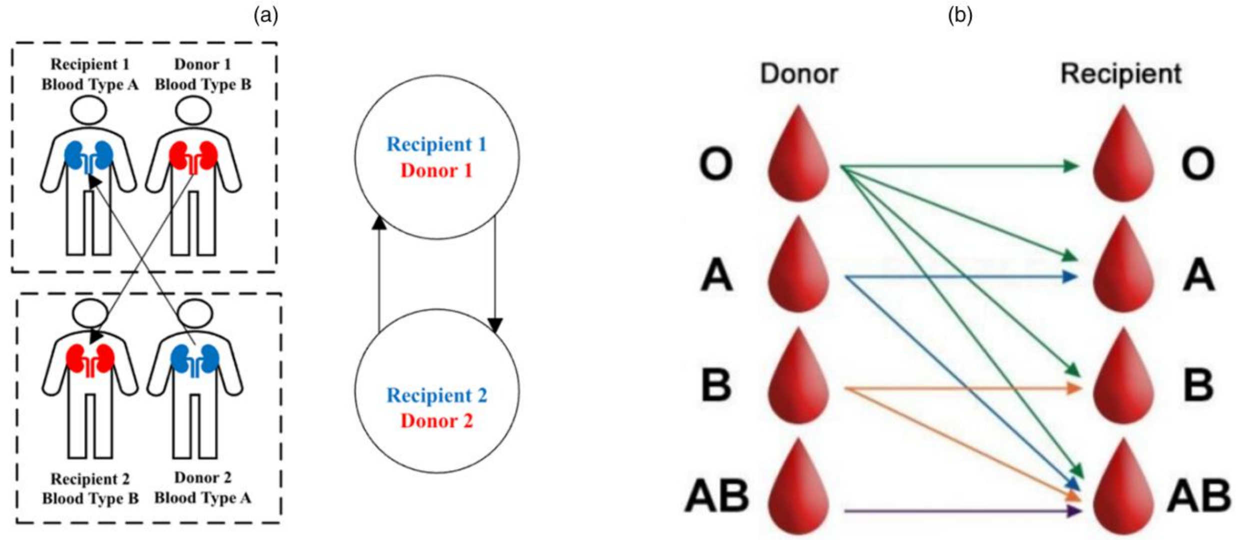
1. Introduction

Chronic kidney disease (CKD) imposes a substantial global mortality burden—about 1.5 million deaths in 2021—while end-stage kidney failure continues to face severe transplant shortages (Deng et al. 2025). In the United States, demand far exceeds supply: roughly 90,000 people are on the kidney waitlist at any time, yet only about 27,000–28,000 transplants were performed in 2023 (National Institute of Diabetes and Digestive and Kidney Diseases 2023). Median waiting times commonly range from three to five years, depending on blood type, geography, and health status (American Kidney Fund 2023). These facts motivate educational activities that illustrate a central lesson of operations research/management science (OR/MS) modeling in kidney exchange: feasibility is not intrinsic, but depends on the compatibility rules and assumptions encoded in the model, which can determine whether exchanges (e.g., certain cycles) are possible at all.

Kidney exchange programs (KEPs)[1] enable donor-recipient pairs to swap donors so that each patient receives a compatible organ. KEPs resolve incompatibilities by *rematching* pairs through cycles and chains so that each patient receives a compatible organ while each willing donor helps another patient (Roth et al. 2004, 2005; Rees et al. 2009). Figure 1 shows a standard two-way exchange alongside a blood-type compatibility chart that underpins many textbook discussions.

The literature frequently illustrates three-way exchanges as a natural extension of two-way swaps, and some expository figures depict such cycles among incompatible pairs (e.g., Delorme et al. 2025, figure 1). Our contribution is not to debate those illustrations in general but to clarify what happens *under an ABO-only compatibility assumption with internally incompatible pairs*, a simplification well suited for a first modeling exercise. Here, *ABO compatibility* refers to the standard blood-type rules (A, B, AB, and O) that determine whether a donor's blood type can safely donate to a recipient's blood type. In practice, kidney exchange programs also account for additional sources of (in)compatibility beyond ABO, such as crossmatch results, tissue

Figure 1. A Two-Way Exchange Between Incompatible Pairs with Blood Types A and B and a Blood-Type Compatibility Diagram for Living Kidney Transplants



matching, and sensitization (e.g., panel reactive antibodies); in this puzzle, we intentionally isolate ABO-only rules to keep the modeling exercise focused and classroom ready.

This paper uses the KEP setting to craft a short, self-contained *feasibility* puzzle for undergraduate operations research courses. Whereas many classroom problems emphasize optimizing an objective, our activity asks a crisp structural question: *Can odd-length cycles—three-way exchanges as the canonical case—occur when every pair is internally incompatible and compatibility is defined solely by ABO blood type?*[2] We note that a cycle is defined over distinct patient-donor pairs, so multiple distinct pairs can share the same ordered blood-type pair; thus, returning to a *pair type* does not close the cycle unless we return to the starting pair. The question is pedagogically appealing because it requires students to translate domain rules into binary assignment and implication constraints, reason carefully about what those constraints permit, and distinguish between "could be optimal" and "is even possible." In operational kidney exchange, ABO blood types are usually observed inputs, and the matching problem selects cycles/chains from a compatibility graph. Here, we instead fix a candidate three-cycle and ask whether ABO-only compatibility—together with internal incompatibility of every pair—can make that pattern feasible.

A growing body of OR/MS education work advocates short, classroom-ready modules that use puzzles, games, and applied contexts to teach modeling—especially integer programming (IP)—in an accessible, hands-on way. Representative examples include puzzle- and game-based IP activities (Chlond 2005, Beliën et al. 2013), board/logic-puzzle modules for core OR ideas (DePuy and Taylor 2007), and IP formulations for popular puzzles that also expose modern modeling concepts (e.g., lazy constraints) (Pearce and Forbes 2017, Hartmann 2018). Related contributions use sports scheduling as a motivating context for teaching integer programming formulations (Goossens and Beliën 2023) and synthesize case-based approaches for OR/MS instruction (Drake 2019). Our puzzle complements this stream by offering a healthcare-motivated *feasibility* activity (kidney exchange) that emphasizes feasibility certificates and presolve-style variable fixings, rather than objective-driven optimization.

We make three practical contributions for instructors and students: (i) we formulate the puzzle as a mixed-integer feasibility model that encodes assignment, incompatibility, and donor-to-next-recipient implication constraints; (ii) we show that simple variable fixings immediately reduce the model and make the feasibility answer transparent, including a direct certificate for (in)feasibility in the three-way case and a template for larger odd cycles; and (iii) we provide a concise Python and Gurobi implementation together with a classroom activity that highlights feasibility reasoning, modular (cyclic) indexing, and the connection between domain rules and binary constraints.

The rest of the paper is organized as follows. Section 2 presents the puzzle statement and visual aid used in class. Section 3 develops the base feasibility formulation and the resulting variable fixings, followed by a reduced model. Section 4 summarizes the Python and Gurobi implementation used to generate certificates. Section 5 briefly reports on our classroom experience and survey instrument.

2. The Puzzle

This section formalizes the classroom activity used in our course. To emphasize scope, the puzzle is a *rule-consistency* exercise: the exchange pattern is fixed (a three-cycle), and the only "decisions" are the missing ABO labels, which are used solely to test whether the ABO-only rules can support such a cycle when all pairs are internally incompatible. Students are given three donor-recipient pairs that are *internally blood incompatible*; in the diagram (Figure 2), each dashed box encloses a donor and intended recipient, and blank labels indicate unknown blood types. A separate blood-compatibility chart for living kidney transplantation appears in Figure 1(b). The goal is to determine whether a three-way exchange can be achieved by assigning ABO types to all blanks while respecting the following rules.

1. Internal incompatibility: within each dashed box, the donor's blood type must be incompatible with the paired recipient's blood type.
2. One-to-one donation: every donor must donate to exactly one recipient.
3. One-to-one reception: every recipient must receive exactly one kidney.

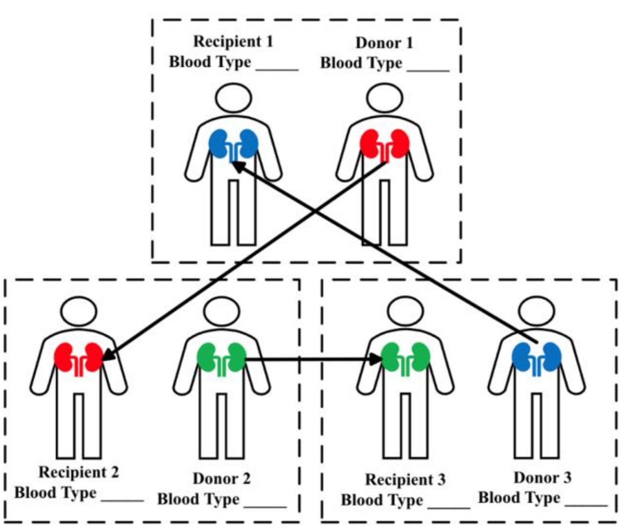
Compatibility on cross-pair arcs must be judged using the chart in Figure 1(b). We restrict attention to ABO-only rules.

2.1. Core Question

Under *ABO-only* compatibility (Figure 1(b)) and assuming *each pair is internally incompatible*, is there an assignment of blood types {A, B, O, AB} to all blanks in Figure 2 that makes the directed three-cycle

$$d_1 \rightarrow r_2, \quad d_2 \rightarrow r_3, \quad d_3 \rightarrow r_1$$

Figure 2. The Three-Way Exchange Puzzle



Note. The color scheme encodes compatibility: similar colors denote blood-type compatibility, and dissimilar colors denote incompatibility.

feasible (i.e., each cross-pair arc is blood compatible while each donor remains incompatible with the paired recipient)?

2.2. Student Tasks

Students model the puzzle as a *feasibility* problem in mixed-integer programming. This structure follows a common puzzle-based modeling approach in OR/MS education: students translate a small set of domain rules into a compact IP feasibility/validation model and then use a standard solver to computationally verify correctness (e.g., Chlond 2005, Pearce and Forbes 2017, Hartmann 2018). A detailed list of tasks is provided below.

- Introduce binary assignment variables for each donor and recipient over {A, B, O, AB}.
- Enforce the three-rule families above (assignment, internal incompatibility, and cycle-compatibility implications) using Figure 1(b).
- Run the model in Python using Gurobi and report either (i) a feasible ABO assignment for Figure 2, or (ii) a clear certificate of infeasibility.

We note that students were responsible for writing the constraints (and their corresponding codes) themselves.

2.3. Deliverables

Students submit (i) a brief statement of the formulation with variables and constraints labeled by role, (ii) code and a concise solver log showing "feasible" or "infeasible," and (iii) a short plain-language explanation tying the outcome to specific constraints.

2.4. Optional Extension

Students are asked to derive a reduced model with fewer variables/constraints by fixing variables implied by the rules (e.g., types that can never appear under internal incompatibility).

The next section provides a clean feasibility formulation that we use in class, followed by simple variable fixings that streamline the model and make the outcome transparent.

3. Formulations and Fixings

We translate the puzzle in Figure 2 into a short mixed-integer *feasibility* model and then apply a few algebraic fixings that expose infeasibility without relying on a solver. Indices are modular: for $i \in \{1,2,3\}$, let $j = 1 + (i \bmod 3)$ denote the next pair in the directed three-cycle.

3.1. Base Formulation

Let $P = \{(r_1, d_1), (r_2, d_2), (r_3, d_3)\}$ be the three recipient-donor pairs. For each pair $i \in \{1,2,3\}$ and blood type $b \in \{A, B, O, AB\}$, define binary variables $x_{r_i,b} = 1$ if recipient r_i has type b and $y_{d_i,b} = 1$ if donor d_i has type

b. We use the (ABO) compatibility function comp(·), where comp(b) is the set of recipient blood types compatible with a donor of type b. Table 1 summarizes ABO compatibility by listing, for each donor type b, the set comp(b) of recipient blood types that can receive from b.

The puzzle rules are encoded as follows.

3.1.1. Rule 1: Assignment. For each pair $i \in \{1,2,3\}$, we have

$$\sum_{b \in \{A,B,O,AB\}} x_{r_i,b} = 1, \qquad \sum_{b \in \{A,B,O,AB\}} y_{d_i,b} = 1. \tag{1}$$

3.1.2. Rule 2: Internal Incompatibility. At each pair $i \in \{1,2,3\}$, the donor d_i is *not* compatible with the paired recipient r_i. So, for each pair $i \in \{1,2,3\}$ and each blood type $b \in \{A,B,O,AB\}$, we have

$$y_{d_i,b} \le 1 - \sum_{\bar{b} \in \text{comp}(b)} x_{r_i,\bar{b}}. \tag{2}$$

Equivalently, one may write the symmetric form $x_{r_i,\bar{b}} \le 1 - \sum_{b:\bar{b} \in \text{comp}(b)} y_{d_i,b}$ for all $i, \bar{b}$; we omit it for brevity.

3.1.3. Rule 3: Cycle Compatibility. At each pair $i \in \{1,2,3\}$, if donor d_i donates to the next-recipient r_j (with $j = 1 + (i \bmod 3)$), then r_j must be compatible with the chosen type of d_i. So, for each blood type $b \in \{A,B,O,AB\}$, for each pair $i \in \{1,2,3\}$, and $j = 1 + (i \bmod 3)$, we have

$$y_{d_i,b} \le \sum_{\bar{b} \in \text{comp}(b)} x_{r_j,\bar{b}}. \tag{3}$$

3.2. Immediate Variable Fixings

Before solving, a simple algebraic observation of the incompatibility rules allows us to eliminate several possibilities.

Fixing 1 ($y_{d_i,O} = 0$ for Every Pair $i \in \{1,2,3\}$). Using Inequality (2) with blood type $b = O$ and its corresponding compatible recipient blood types comp(O) = {O, A, B, AB} gives

$$\begin{aligned} y_{d_i,O} &\le 1 - \sum_{\bar{b} \in \text{comp}(O)} x_{r_i,\bar{b}} \\ &= 1 - \sum_{b \in \{A,B,O,AB\}} x_{r_i,b} = 0, \end{aligned}$$

where the last equality follows from (1).

Table 1. ABO Compatibility: Recipient Types Compatible with a Given Donor Type

Donor type (b)	Compatible recipient types (comp(b))
O	{O, A, B, AB}
A	{A, AB}
B	{B, AB}
AB	{AB}

Fixing 2 ($x_{r_i,AB} = 0$ for Every Pair $i \in \{1,2,3\}$). A recipient of type AB is compatible with every donor type (AB is the universal recipient), so within-box incompatibility forces $r_i \ne$ AB; hence, $x_{r_i,AB} = 0$ for every pair $i \in \{1,2,3\}$.

Fixing 3 ($y_{d_i,AB} = 0$ for Every Pair $i \in \{1,2,3\}$). From (3) with b = AB and comp(AB) = {AB}, we have

$$y_{d_i,AB} \le x_{r_j,AB} = 0,$$

using Fixing 2.

Fixing 4. ($x_{r_i,O} = 0$ for Every Pair $i \in \{1,2,3\}$). In the ABO rules, the only donor type compatible with O is O itself. Thus, if $x_{r_j,O} = 1$, cycle compatibility would require $y_{d_i,O} = 1$ for the predecessor donor, contradicting Fixing 1. Hence, we have $x_{r_j,O} = 0$ for every pair $j \in \{1,2,3\}$.

After applying Fixing 1–Fixing 4, only blood types A and B remain for the involved donors and recipients.

3.3. Reduced Formulation (A/B Only)

Restricting each donor and recipient to blood types A and B yields a two-type feasibility system. The assignment rule (rule 1) becomes a one-of-two assignment. The internal incompatibility rule (rule 2) becomes "donor and recipient differ" in each pair. The cycle compatibility rule (rule 3) restricts donations within {A, B}: an A-donor may donate only to an A-recipient and a B-donor only to a B-recipient. So, our original formulation is reduced as follows. For every pair $i \in \{1,2,3\}$, we have

$$x_{r_i,A} + x_{r_i,B} = 1, \qquad y_{d_i,A} + y_{d_i,B} = 1 \tag{4a}$$

$$y_{d_i,A} \le 1 - x_{r_i,A}, \qquad y_{d_i,B} \le 1 - x_{r_i,B}. \tag{4b}$$

Furthermore, for every pair $i \in \{1,2,3\}$ and $j = 1 + (i \bmod 3)$, we have

$$y_{d_i,A} = x_{r_j,A}, \qquad y_{d_i,B} = x_{r_j,B}. \tag{4c}$$

The infeasibility is now immediate: (4c) propagates recipient labels around the three-cycle (forcing all recipients to share the same A/B label), whereas (4b) forces a flip between donor and recipient within each pair; an odd cycle cannot satisfy simultaneous "propagate" and "flip" requirements.

4. Solution

We implement the feasibility model in Python using the Gurobi Optimizer (`gurobipy`) and solve either for a valid ABO assignment that realizes the three-cycle or for a certificate of infeasibility (Gurobi Optimization, LLC 2025). The code mirrors the formulation in Section 3: it constructs binary assignment variables for each donor and recipient over {A, B, O, AB}, adds the assignment constraints, encodes internal (within-pair) incompatibility, and enforces donor-to-next-recipient

Table 2. Postactivity Likert-Scale Survey

Statement	1	2	3	4	5
The activity improved my skill in translating real-world puzzles into optimization models.	0	1	3	21	15
I can explain how blood compatibility rules translate into mathematical constraints.	0	2	8	17	13
Implementing the model in Python+Gurobi clarified how a mathematical formulation can be programmed.	0	0	6	27	7
I felt more engaged with the course material by participating in this activity.	0	1	10	16	13
The activity increased my interest in OR applications in healthcare (e.g., kidney exchange).	0	8	9	7	16
The instructions, time allocation, and materials were appropriate for the course level.	1	0	8	16	15
Similar puzzle-based activities should be used in future IE 3311 offerings.	0	1	9	13	17
The activity improved my problem-solving and analytical thinking skills.	0	1	3	23	13

Note. Response scale: 1 = strongly disagree, 2 = disagree, 3 = neutral, 4 = agree, 5 = strongly agree.

implications using modular indexing $j = 1 + (i \bmod k)$ to express the directed cycle succinctly. For transparency in class, each block of constraints is created by a short function whose name matches the role used in the paper.

For classroom use, the model solves essentially instantly on a standard laptop. We encourage students to echo the counts of variables and constraints before and after the elementary fixings from Section 3 to visualize how presolve-style reasoning shrinks the model and to save a concise solver log showing the final status (FEASIBLE/INFEASIBLE). A minimal and self-contained coding package is provided on GitHub.[3]

5. Classroom Experience

We implemented the activity in an undergraduate operations research class. Forty undergraduate students completed the puzzle. Before the activity, students had covered basic binary IP modeling and had completed prior Python and Gurobi exercises; they were also familiar with the kidney-exchange setting at a conceptual level. The in-class workflow was simple and deliberate: students first read the KEP context and the three rules, then translated those rules into a compact mixed-integer feasibility model, and finally tested their formulations in Python and Gurobi. Working individually, they were asked to label each constraint by role (assignment, incompatibility, implication) and to explain—in two or three sentences—why their model either produced a valid assignment or certified infeasibility. This emphasis on labeled constraints and short plain-language explanations helped keep the focus on feasibility reasoning rather than solver "tuning."

The activity is intentionally "active learning" oriented: students must construct and test a formulation, diagnose infeasibility, and articulate a brief certificate rather than only follow a worked example. This design choice is consistent with evidence that active-learning approaches improve student performance in STEM settings relative to lecture-only formats (e.g., Prince 2004, Freeman et al. 2014).

To capture perceptions and learning outcomes, we administered a short, anonymous survey immediately after the submission deadline. The instrument consisted of Likert-scale statements on a five-point scale (strongly disagree to strongly agree) (Likert 1932) targeting (i) conceptual understanding of feasibility versus optimization, (ii) the translation of medical rules into binary constraints, (iii) confidence with modular indexing on cycles, and (iv) engagement and course relevance. The exact prompts used in class appear in Table 2. We note that 40 students participated in the survey.

Instructors reported that the activity fits comfortably within a single class meeting, including a short debrief. A practical pacing guide is 10–12 minutes for context and rules, 30–35 minutes for modeling and coding, and 5–8 minutes for a wrap-up that contrasts feasibility certificates with objective-driven models and highlights how simple variable fixings (Section 3) clarify the outcome. Keeping the code modular—one short function per constraint family—made it straightforward for students to trace each line of their formulation to a handful of implementation lines and to diagnose infeasibility using Gurobi's output. The survey responses (and informal feedback) suggest that the puzzle is an effective way to connect domain rules, modeling, and computational verification in a compact, undergraduate-friendly format.

Endnotes

[1] They are also referred to as kidney paired donation (KPD) programs.

[2] Under the same assumptions, even-length cycles are not ruled out; in fact, the reduced structure permits only even directed cycles, so the infeasibility result is specific to odd lengths.

[3] See (i) an instructor-facing repository at https://github.com/gabrieltepin/odd_cycle_puzzle/tree/master, which contains the Python scripts as well as a comprehensive `README`; and (ii) a lightweight student-facing interface at https://gabrieltepin.github.io/odd_cycle_puzzle.

References

American Kidney Fund (2023) Kidney transplant waiting list. Accessed June 8, 2026, https://www.kidneyfund.org/kidney-donation-and-transplant/transplant-waiting-list.

Beliën J, Colpaert J, De Boeck L, Eyckmans J, Leirens S (2013) Teaching integer programming starting from an energy supply game. *INFORMS Trans. Ed.* 13(3):129–137.

Chlond M (2005) Classroom exercises in IP modeling: Su Doku and the Log Pile. *INFORMS Trans. Ed.* 5(2):77–79.

Delorme M, Liu W, Manlove D (2025) Mathematical models and exact algorithms for kidney exchange problems with immunosuppressants. *INFORMS J. Comput.*, ePub ahead of print December 29, https://doi.org/10.1287/ijoc.2024.1071.

Deng L, Guo S, Liu Y, Zhou Y, Liu Y, Zheng X, Yu X, Shuai P (2025) Global, regional, and national burden of chronic kidney disease and its underlying etiologies from 1990 to 2021: A systematic analysis for the Global Burden of Disease Study 2021. *BMC Public Health* 25, article 636.

DePuy GW, Taylor GD (2007) Using board puzzles to teach operations research. *INFORMS Trans. Ed.* 7(2):160–171.

Drake MJ (2019) Teaching OR/MS with cases: A review and new suggestions. *INFORMS Trans. Ed.* 19(2):57–66.

Freeman S, Eddy SL, McDonough M, Smith MK, Okoroafor N, Jordt H, Wenderoth MP (2014) Active learning increases student performance in science, engineering, and mathematics. *Proc. Natl. Acad. Sci. USA* 111(23):8410–8415.

Goossens D, Beliën J (2023) Teaching integer programming by scheduling the Belgian soccer league. *INFORMS Trans. Ed.* 23(3):164–172.

Gurobi Optimization, LLC (2025) Gurobi optimizer reference manual. Accessed June 8, 2026, https://docs.gurobi.com/projects/optimizer/en/current/index.html.

Hartmann S (2018) Puzzle—Solving smartphone puzzle apps by mathematical programming. *INFORMS Trans. Ed.* 18(2):127–141.

Likert R (1932) *A Technique for the Measurement of Attitudes*, Archives of Psychology, No. 140 (The Science Press, New York), 1–55.

National Institute of Diabetes and Digestive and Kidney Diseases (2023) Kidney disease statistics for the United States. Accessed June 8, 2026, https://www.niddk.nih.gov/health-information/health-statistics/kidney-disease.

Pearce J, Forbes MA (2017) Puzzle—Fillomino: An integer programming model with lazy constraints. *INFORMS Trans. Ed.* 17(2):85–89.

Prince M (2004) Does active learning work? A review of the research. *J. Engrg. Ed.* 93(3):223–231.

Rees MA, Kopke JE, Pelletier RP, Segev DL, Rutter ME, Fabrega AJ, Rogers J, et al. (2009) A nonsimultaneous, extended, altruistic-donor chain. *N England J. Medicine* 360(11):1096–1101.

Roth AE, Sönmez T, Ünver MU (2004) Kidney exchange. *Quart. J. Econom.* 119(2):457–488.

Roth AE, Sönmez T, Ünver MU (2005) Pairwise kidney exchange. *J. Econom. Theory* 125(2):151–188.