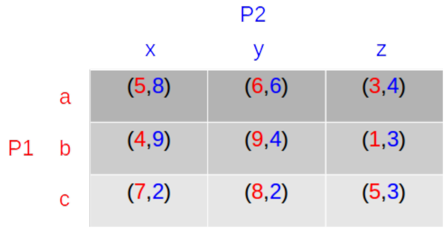
Problem 1. (20 points)

For the following payoff matrix, consider only pure strategies and find:



		P2		
		x	y	z
	a	(5,8)	(6,6)	(3,4)
P1	b	(4,9)	(9,4)	(1,3)
	c	(7,2)	(8,2)	(5,3)

Figure 1: Provided payoff matrix.

(1) **All Nash equilibrium**

To find all pure strategy Nash equilibrium, consider each possible pair of strategies individually:

(a,x): Not a Nash equilibrium since P1 prefers strategy c in response to P2 choosing strategy x.
(a,y): Not a Nash equilibrium since P1 prefers strategy b in response to P2 choosing strategy y.
(a,z): Not a Nash equilibrium since P1 preferes strategy c in response to P2 choosing strategy z.
(b,x): Not a Nash equilibrium since P1 prefers strategy c in response to P2 choosing strategy x.
(b,y): Not a Nash equilibrium since P2 prefers strategy x in response to P1 choosing strategy b.
(b,z): Not a Nash equilibrium since P1 prefers strategy c in response to P2 choosing strategy z.
(c,x): Not a Nash equilibrium since P2 prefers strategy c in response to P1 choosing strategy c.
(c,y): Not a Nash equilibrium since P1 prefers strategy b in response to P2 choosing strategy y.
(c,z): Is a Nash equilibrium since both players have no other strategy that does better at maximizing their payoff in response to their opponents selected strategy.

Therefore, (c,z) is the only pure strategy Nash equilibrium for the given payoff matrix.

(2) **All Pareto optimal strategies**

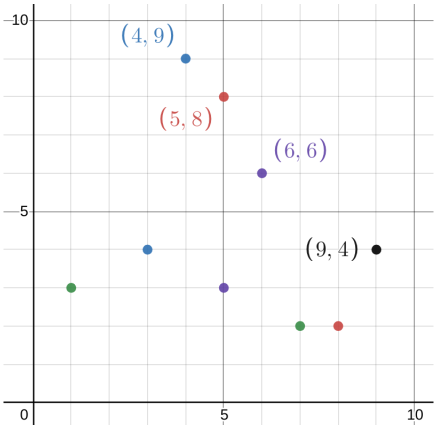


Figure 2: Plot of all nine strategy pairs from the provided payoff matrix.

Notice that only the labeled points do not have any other points which fall into the region of the graph which would be called quadrant I (including values lying on the positive Y or positive X axes) if the point in question were the origin. This means that the corresponding strategy pairs (b,x), (a,x), (a,y), and (b,y) comprise all Pareto optimal strategies, since for each of these four strategy pairs, no other strategy pair improves the payout received by one player without decreasing the payout received by the other player.

In summary, (b,x), (a,x), (a,y), and (b,y) constitute all Pareto optimal pure strategies.

Problem 2. (20 points)

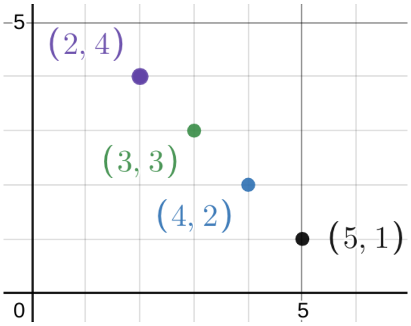
Suppose you had a 2-by-2 payoff matrix where all payoffs are zero-sum (to the same constant value). Additionally, each payoff value cannot be duplicated. For pure strategies, what are:

(1) **The maximum number of Pareto optimal strategies**

First, note that it is immediately apparent that there cannot be more than four Pareto optimal pure strategies since there are only four possible strategy pairs for a 2-by-2 payoff matrix. As such, if we can find any 2-by-2 payoff matrix–where all payoffs are zero-sum, and no payoffs are duplicated–with four Pareto optimal pure strategies, we will have confirmed that the maximum number of Pareto optimal pure strategies is exactly four. Observe the following payoff matrix and plot:

	x	y
a	(2,4)	(3,3)
b	(4,2)	(5,1)

(a)

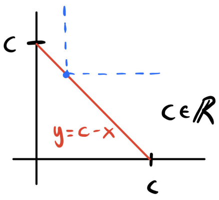


(b)

Hence, each strategy pair is Pareto optimal as evidenced by plot (b), and all payoffs in the matrix are zero-sum ($c = 6$), with no duplicated payoff pairs. Therefore, we have shown that there exists at least one 2-by-2 zero-sum payoff matrix which has 4 different Pareto optimal pure strategies. Therefore, the maximum number of Pareto optimal pure strategies for the described type of payoff matrix in necessarily 4 since it is not possible for there to be more (for the reasons described above).

(2) **The minimum number of Pareto optimal strategies**

First, note that each payoff pair (x_i,y_i), $i \in [0, 3]$ must satisfy $x_i + y_i = c$ since all payoffs are zero-sum. It follows that all 4 payoff pairs must lie somewhere on the line $y = c - x$. Additionally, for any $i, j \in [0, 3], i \neq j$, we have that $(x_i, y_i) \neq (x_j, y_j)$ since payoff values cannot be duplicated. As such, all four payoff pairs will necessarily be Pareto optimal strategies since the slope of $y = c - x$ is -1 (meaning that for any point on the line there are no points which fall into the region of the graph which would be called quadrant I (including values lying on the positive Y or positive X axes) if the point in question were the origin:



Hence, the minimum number of Pareto optimal pure strategies is 4.

Continued on next page ⟶

(3) The maximum number of Nash equilibrium

Observe the following generalization of zero-sum, 2-by-2 payoff matrices without duplicates (i.e. $a \neq c \neq d \neq e$):

	x	y
f	(a,b)	(c,a+b-c)
g	(d,a+b-d)	(e,a+b-e)

It follows that for (f,x) to be a Nash equilibrium, the following condition must hold:

$$\begin{aligned} & a \geq d \texttt{ AND } b \geq a+b-c \\ \iff & a \geq d \texttt{ AND } c \geq a \\ \iff & a > d \texttt{ AND } c > a \quad \texttt{// since } a \neq c \neq d \neq e \end{aligned}$$

For (f,y) to be a Nash equilibrium, the following condition must hold:

$$\begin{aligned} & c \geq e \texttt{ AND } a+b-c \geq b \\ \iff & c \geq e \texttt{ AND } a \geq c \\ \iff & c > e \texttt{ AND } a > c \quad \texttt{// since } a \neq c \neq d \neq e \end{aligned}$$

For (g,x) to be a Nash equilibrium, the following condition must hold:

$$\begin{aligned} & d \geq a \texttt{ AND } a+b-d \geq a+b-e \\ \iff & d \geq a \texttt{ AND } e \geq d \\ \iff & d > a \texttt{ AND } e > d \quad \texttt{// since } a \neq c \neq d \neq e \end{aligned}$$

For (g,y) to be a Nash equilibrium, the following condition must hold:

$$\begin{aligned} & e \geq c \texttt{ AND } a+b-e \geq a+b-d \\ \iff & e \geq c \texttt{ AND } d \geq e \\ \iff & e > c \texttt{ AND } d > e \quad \texttt{// since } a \neq c \neq d \neq e \end{aligned}$$

First, consider if it is possible for all four payoff pairs to be Nash equilibriums. For this to be true, the following must hold for some a,c,d,e such that $a \neq c \neq d \neq e$:

$$(a > d \texttt{ AND } c > a) \texttt{ AND } (c > e \texttt{ AND } a > c) \texttt{ AND } (d > a \texttt{ AND } e > d) \texttt{ AND } (e > c \texttt{ AND } d > e)$$

This is impossible as you cannot have $c > a$ and $c < a$, among other disqualifying conditions.

Next, consider if it is possible for three payoff pairs to be Nash equilibriums. For this to be true, any of the following 4 (4 choose 3) expressions must hold for some a,c,d,e such that $a \neq c \neq d \neq e$:

$$(a > d \texttt{ AND } c > a) \texttt{ AND } (c > e \texttt{ AND } a > c) \texttt{ AND } (d > a \texttt{ AND } e > d)$$

$$(a > d \texttt{ AND } c > a) \texttt{ AND } (c > e \texttt{ AND } a > c) \texttt{ AND } (e > c \texttt{ AND } d > e)$$

These are both impossible as you cannot have $c > a$ and $c < a$.

Continued on next page ⟶

$$(a > d \texttt{ AND } c > a) \texttt{ AND } (d > a \texttt{ AND } e > d) \texttt{ AND } (e > c \texttt{ AND } d > e)$$

$$(c > e \texttt{ AND } a > c) \texttt{ AND } (d > a \texttt{ AND } e > d) \texttt{ AND } (e > c \texttt{ AND } d > e)$$

These are both impossible as you cannot have $e > d$ and $e < d$.

Next consider if it is possible to have 2 payoff pairs be Nash equilibriums. For this to be true, any of the 6 (4 choose 2) possible combinations of $(a > d \texttt{ AND } c > a), (c > e \texttt{ AND } a > c), (d > a \texttt{ AND } e > d), (e > c \texttt{ AND } d > e)$ must hold for some a,c,d,e such that $a \neq c \neq d \neq e$:

$(a > d \texttt{ AND } c > a) \texttt{ AND } (c > e \texttt{ AND } a > c)$ // impossible b/c $c > a$ and $a > c$ can't happen

$(a > d \texttt{ AND } c > a) \texttt{ AND } (d > a \texttt{ AND } e > d)$ // impossible b/c $a > d$ and $d > a$ can't happen

$(a > d \texttt{ AND } c > a) \texttt{ AND } (e > c \texttt{ AND } d > e)$ // impossible b/c $(d > e > c > a > d)$ can't happen

$(c > e \texttt{ AND } a > c) \texttt{ AND } (d > a \texttt{ AND } e > d)$ // impossible b/c $(e > d > a > c > e)$ can't happen

$(c > e \texttt{ AND } a > c) \texttt{ AND } (e > c \texttt{ AND } d > e)$ // impossible b/c $c > e$ and $e > c$ can't happen

$(d > a \texttt{ AND } e > d) \texttt{ AND } (e > c \texttt{ AND } d > e)$ // impossible b/c $e > d$ and $d > e$ can't happen

Therefore, it is not possible to have 4, 3, or 2 pure strategy Nash equilibriums for the described class of payoff matrices.

Finally, consider if it is possible to have 1 payoff pair be a Nash equilibrium. For this to be true, any one of $(a > d \texttt{ AND } c > a), (c > e \texttt{ AND } a > c), (d > a \texttt{ AND } e > d), (e > c \texttt{ AND } d > e)$ must hold for some a,c,d,e such that $a \neq c \neq d \neq e$. In fact, all of these individual statements are true for some a,c,d,e such that $a \neq c \neq d \neq e$ when not combined. For example, using $(a > d \texttt{ AND } c > a)$, corresponding to strategy pair (f,x) being a Nash equilibrium, let d = 1, a = 2, c = 3, e = 4, and b = 0. $(a > d \texttt{ AND } c > a)$ is satisfied, and the resulting payoff matrix:

	x	y
f	(2,0)	(3,-1)
g	(1,1)	(4,-2)

does, in fact, have only one pure strategy Nash equilibrium (f,x).

Therefore, the maximum number of pure strategy Nash equilibrium for a zero-sum, 2-by-2 payoff matrix without duplicate payoff pairs is 1.

(4) **The minimum number of Nash equilibrium**

Note that it is not possible for there to be fewer than zero Nash equilibrium. Thus, if we can find any 2-by-2 payoff matrix with 0 pure strategy Nash equilibriums, we will have shown that the minimum number of Nash equilibrium is necessarily 0. Observe the following zero-sum ($c = 1$) payoff matrix without duplicate payoff pairs:

	x	y
a	(0,1)	(3,-2)
b	(4,-3)	(2,-1)

Considering each strategy pair, we see that there are no pure strategy Nash equilibriums:

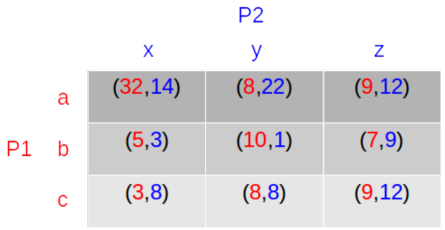
(a,x): Not a Nash equilibrium since the row player prefers strategy b if the column player chooses strategy x.
(a,y): Not a Nash equilibrium since the column player prefers strategy x if the row player chooses strategy a.
(b,x): Not a Nash equilibrium since the column player prefers strategy y if the row player chooses strategy b.
(b,y): Not a Nash equilibrium since the row player prefers strategy a if the column player chooses strategy y.

Continued on next page ⟶

Therefore, there exists at least one 2-by-2, zero-sum payoff matrix without duplicate payoff pairs which has the 0 pure strategy Nash equilibrium, meaning that the minimum number of pure strategy Nash equilibrium for the described class of payout matrices is necessarily 0 (since there cannot be fewer than 0).

Problem 3. (20 points)

Consider the following 3-by-3 payoff matrix.



		P2		
		x	y	z
	a	(32,14)	(8,22)	(9,12)
P1	b	(5,3)	(10,1)	(7,9)
	c	(3,8)	(8,8)	(9,12)

Figure 4: Provided payoff matrix.

For mixed strategies find:

(1) **At least one Nash equilibrium**

Consider the following 2-by-2 sub matrix:

	y	z
a	(8,22)	(9,12)
b	(10,1)	(7,9)

Let P1 choose strategy a with a probability of p and strategy b with a probability of 1-p. It follows that P2's expected payout is $(p)(22) + (1-p)(1) = 21p + 1$ if they choose strategy y and $(p)(12) + (1-p)(9) = 3p + 9$ if they choose strategy z. Setting these two equations equal and solving yields:

$$21p + 1 = 3p + 9 \iff 18p = 8 \iff p = \frac{4}{9}$$

Next let P2 choose strategy y with a probability of q and strategy z with a probability of 1-q. It follows that P1's expected payout is $(q)(8) + (1-q)(9) = 9 - q$ if they choose strategy a and $(q)(10) + (1-q)(7) = 7 + 3q$ if they choose strategy b. Setting these two equations equal and solving yields:

$$9 - q = 7 + 3q \iff 2 = 4q \iff q = \frac{1}{2}$$

Hence, the mixed strategy Nash equilibrium to this sub problem is that P1 chooses strategy a, $\frac{4}{9}$ of the time and strategy b, $\frac{5}{9}$ of the time, while P2 chooses strategy y, $\frac{1}{2}$ of the time and strategy z, $\frac{1}{2}$ of the time. The expected payout for P1 is therefore $\frac{1}{2} * 8 + \frac{1}{2} * 9 = 8.5$ and the expected payout for P2 is $\frac{4}{9} * 22 + \frac{5}{9} * 1 = \frac{31}{3}$.

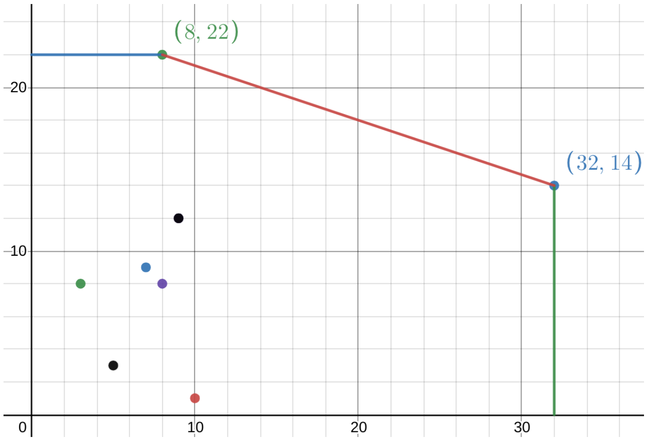
Returning to the full 3-by-3 payout matrix. We see that indeed, if P1 chooses strategy a, $\frac{4}{9}$ of the time and strategy b, $\frac{5}{9}$ of the time, P2's expected payout if they choose strategy x is $\frac{4}{9} * 14 + \frac{5}{9} * 3 = \frac{71}{9} < \frac{31}{3}$. Hence, P2 does not wish to change strategies. Additionally, if P2 chooses strategy y, $\frac{1}{2}$ of the time and strategy z, $\frac{1}{2}$ of the time, P1's expected payout if they choose strategy c is $\frac{1}{2} * 8 + \frac{1}{2} * 9 = 8.5$ which is no better than their current expected payout. Hence, P1 has nothing to gain by changing strategies either.

Therefore, the mixed strategy where P1 chooses strategies, a, b, and c with probabilities of $\frac{4}{9}$,$\frac{5}{9}$, and 0 respectively, and P2 chooses strategies x,y, and z with probabilities of 0, $\frac{1}{2}$, and $\frac{1}{2}$ respectively, is a Nash equilibrium.

Continued on next page ⟶

(2) **All Pareto optimal strategies**

Observe the following plot of all 9 payout pairs from the provided payoff matrix



Any point along the red line segment $y = \frac{74}{3} - \frac{1}{3}x; 8 \leq x \leq 32$ corresponds to a Pareto optimal strategy. Note that the payout pair (8,22) corresponds to P1 choosing strategy `a` and P2 choosing strategy `y`, while the payout pair (32,14) corresponds to P1 choosing strategy `a` and P2 choosing strategy `x`. Hence, the following describes all Pareto optimal strategies: For any `p` such that $0 \leq p \leq 1$, P2 chooses strategy `x` with probability `p` and strategy `y` with probability `1-p`, and P1 chooses strategy `a` every time.

Problem 4. (20 points)

For the problem below, show for k-consistency:

w, x, y, and z are all single digit integers (0-9)

$w < x < y < z$

$w + x > z$

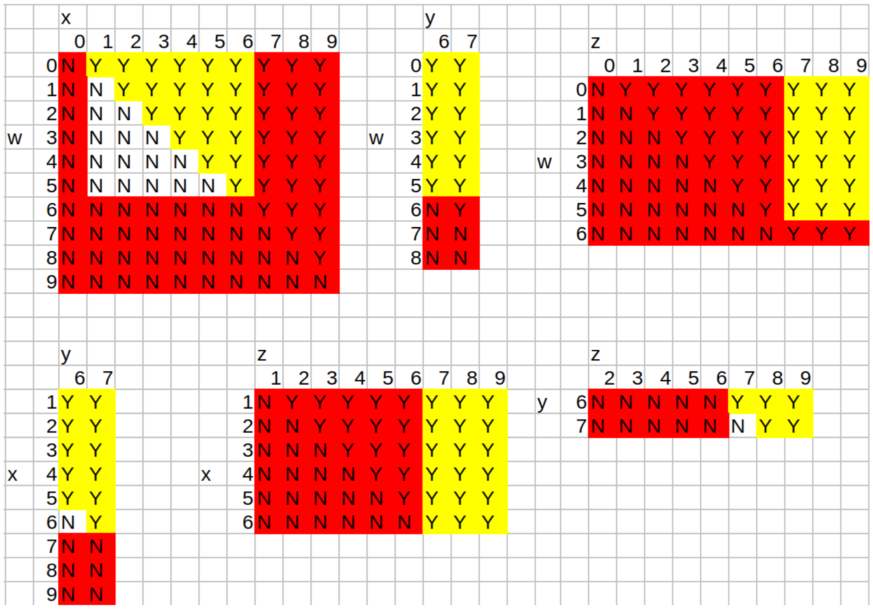
$5 < y < 8$

$z - y = x - w$

(1) **The domains of variables that are 1-consistent**

$\mathcal{D}_w = \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9\}$
$\mathcal{D}_x = \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9\}$
$\mathcal{D}_y = \{6, 7\}$ // since $5 < y < 8$
$\mathcal{D}_z = \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9\}$

(2) **The domains of variables that are 2-consistent (using part (1) as a starting point)**



		x									
		0	1	2	3	4	5	6	7	8	9
	0	N	Y	Y	Y	Y	Y	Y	Y	Y	Y
	1	N	N	Y	Y	Y	Y	Y	Y	Y	Y
	2	N	N	N	Y	Y	Y	Y	Y	Y	Y
w	3	N	N	N	N	Y	Y	Y	Y	Y	Y
	4	N	N	N	N	N	Y	Y	Y	Y	Y
	5	N	N	N	N	N	N	Y	Y	Y	Y
	6	N	N	N	N	N	N	N	Y	Y	Y
	7	N	N	N	N	N	N	N	N	Y	Y
	8	N	N	N	N	N	N	N	N	N	Y
	9	N	N	N	N	N	N	N	N	N	N

		y	
		6	7
	0	Y	Y
	1	Y	Y
	2	Y	Y
w	3	Y	Y
	4	Y	Y
	5	Y	Y
	6	N	Y
	7	N	N
	8	N	N

		z									
		0	1	2	3	4	5	6	7	8	9
	0	N	Y	Y	Y	Y	Y	Y	Y	Y	Y
	1	N	N	Y	Y	Y	Y	Y	Y	Y	Y
	2	N	N	N	Y	Y	Y	Y	Y	Y	Y
w	3	N	N	N	N	Y	Y	Y	Y	Y	Y
	4	N	N	N	N	N	Y	Y	Y	Y	Y
	5	N	N	N	N	N	N	Y	Y	Y	Y
	6	N	N	N	N	N	N	N	Y	Y	Y

		y	
		6	7
	1	Y	Y
	2	Y	Y
	3	Y	Y
x	4	Y	Y
	5	Y	Y
	6	N	Y
	7	N	N
	8	N	N
	9	N	N

		z								
		1	2	3	4	5	6	7	8	9
	1	N	Y	Y	Y	Y	Y	Y	Y	Y
	2	N	N	Y	Y	Y	Y	Y	Y	Y
	3	N	N	N	Y	Y	Y	Y	Y	Y
x	4	N	N	N	N	Y	Y	Y	Y	Y
	5	N	N	N	N	N	Y	Y	Y	Y
	6	N	N	N	N	N	N	Y	Y	Y

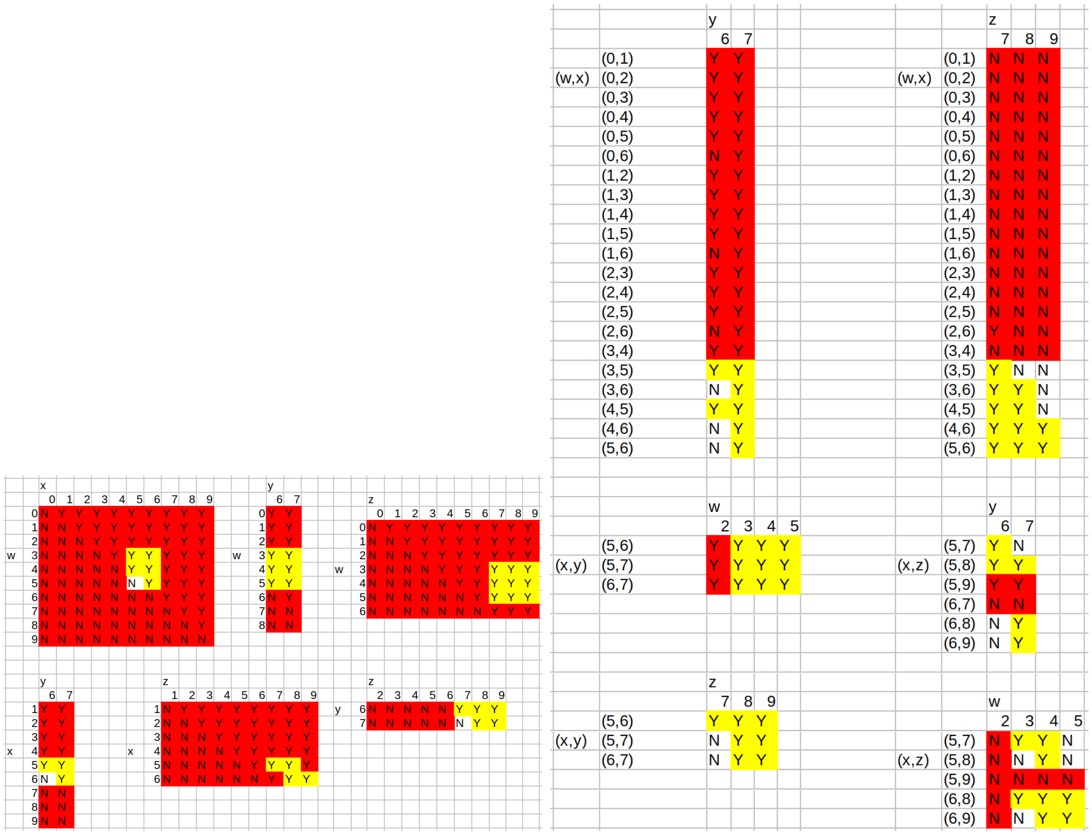
		z							
		2	3	4	5	6	7	8	9
y	6	N	N	N	N	N	Y	Y	Y
	7	N	N	N	N	N	N	Y	Y

Figure 5: Tables used to determine domains of 2-consistent variables

$\mathcal{D}_w = \{0, 1, 2, 3, 4, 5\}$
$\mathcal{D}_x = \{1, 2, 3, 4, 5, 6\}$
$\mathcal{D}_y = \{6, 7\}$
$\mathcal{D}_z = \{7, 8, 9\}$

Continued on next page ⟶

(3) **Of the pairs that involve "x", which values are 3-consistent (using part (2) as a starting point) (Note: only need to show pairs with "x" to reduce work, not due to any CSP related reason.)**



		x									
		0	1	2	3	4	5	6	7	8	9
	0	N	Y	Y	Y	Y	Y	Y	Y	Y	Y
	1	N	N	Y	Y	Y	Y	Y	Y	Y	Y
	2	N	N	N	Y	Y	Y	Y	Y	Y	Y
w	3	N	N	N	N	Y	Y	Y	Y	Y	Y
	4	N	N	N	N	N	Y	Y	Y	Y	Y
	5	N	N	N	N	N	N	Y	Y	Y	Y
	6	N	N	N	N	N	N	N	Y	Y	Y
	7	N	N	N	N	N	N	N	N	Y	Y
	8	N	N	N	N	N	N	N	N	N	Y
	9	N	N	N	N	N	N	N	N	N	N

		y	
		6	7
	0	Y	Y
	1	Y	Y
	2	Y	Y
w	3	Y	Y
	4	Y	Y
	5	Y	Y
	6	N	Y
	7	N	N
	8	N	N

		z									
		0	1	2	3	4	5	6	7	8	9
	0	N	Y	Y	Y	Y	Y	Y	Y	Y	Y
	1	N	N	Y	Y	Y	Y	Y	Y	Y	Y
	2	N	N	N	Y	Y	Y	Y	Y	Y	Y
w	3	N	N	N	N	Y	Y	Y	Y	Y	Y
	4	N	N	N	N	N	Y	Y	Y	Y	Y
	5	N	N	N	N	N	N	Y	Y	Y	Y
	6	N	N	N	N	N	N	N	Y	Y	Y

		y	
		6	7
	1	Y	Y
	2	Y	Y
	3	Y	Y
x	4	Y	Y
	5	Y	Y
	6	N	Y
	7	N	N
	8	N	N
	9	N	N

		z								
		1	2	3	4	5	6	7	8	9
	1	N	Y	Y	Y	Y	Y	Y	Y	Y
	2	N	N	Y	Y	Y	Y	Y	Y	Y
	3	N	N	N	Y	Y	Y	Y	Y	Y
x	4	N	N	N	N	Y	Y	Y	Y	Y
	5	N	N	N	N	N	Y	Y	Y	Y
	6	N	N	N	N	N	N	Y	Y	Y

		z							
		2	3	4	5	6	7	8	9
y	6	N	N	N	N	N	Y	Y	Y
	7	N	N	N	N	N	N	Y	Y

		y	
		6	7
	(0,1)	Y	Y
(w,x)	(0,2)	Y	Y
	(0,3)	Y	Y
	(0,4)	Y	Y
	(0,5)	Y	Y
	(0,6)	N	Y
	(1,2)	Y	Y
	(1,3)	Y	Y
	(1,4)	Y	Y
	(1,5)	Y	Y
	(1,6)	N	Y
	(2,3)	Y	Y
	(2,4)	Y	Y
	(2,5)	Y	Y
	(2,6)	N	Y
	(3,4)	Y	Y
	(3,5)	Y	Y
	(3,6)	N	Y
	(4,5)	Y	Y
	(4,6)	N	Y
	(5,6)	N	Y

		z		
		7	8	9
	(0,1)	N	N	N
(w,x)	(0,2)	N	N	N
	(0,3)	N	N	N
	(0,4)	N	N	N
	(0,5)	N	N	N
	(0,6)	N	N	N
	(1,2)	N	N	N
	(1,3)	N	N	N
	(1,4)	N	N	N
	(1,5)	N	N	N
	(1,6)	N	N	N
	(2,3)	N	N	N
	(2,4)	N	N	N
	(2,5)	N	N	N
	(2,6)	Y	N	N
	(3,4)	N	N	N
	(3,5)	Y	N	N
	(3,6)	Y	Y	N
	(4,5)	Y	Y	N
	(4,6)	Y	Y	Y
	(5,6)	Y	Y	Y

		w			
		2	3	4	5
	(5,6)	Y	Y	Y	Y
(x,y)	(5,7)	Y	Y	Y	Y
	(6,7)	Y	Y	Y	Y

		y	
		6	7
	(5,7)	Y	N
(x,z)	(5,8)	Y	Y
	(5,9)	Y	Y
	(6,7)	N	N
	(6,8)	N	Y
	(6,9)	N	Y

		z		
		7	8	9
	(5,6)	Y	Y	Y
(x,y)	(5,7)	N	Y	Y
	(6,7)	N	Y	Y

		w			
		2	3	4	5
	(5,7)	N	Y	Y	N
(x,z)	(5,8)	N	N	Y	N
	(5,9)	N	N	N	N
	(6,8)	N	Y	Y	Y
	(6,9)	N	N	Y	Y

Figure 6: Tables used to determine domains of 3-consistent variables using only the pairs that involve "x"

$\mathcal{D}_w = \{3, 4, 5\}$
$\mathcal{D}_x = \{5, 6\}$
$\mathcal{D}_y = \{6, 7\}$
$\mathcal{D}_z = \{7, 8, 9\}$

Problem 5. (20 points)

For the problem below, suppose we were doing backtracking search

w, x, y, and z are all single digit integers (0-9)

$w < x < y < z$

$w + x > z$

$5 < y < 8$

$z - y = x - w$

(1) **If we are following the four recommendations for backtracking search, what variable should be picked first?**

Per the recommendations for backtracking search, the variable with the smallest domain size should be picked first, and, in the event of ties, the variable with the most connections should be chosen. For this problem, before any search or inference has been done, the domains sizes of all of the variables is the same. As such, the degree heuristic is needed to break the variable selection tie. As indicated in figure 7, variable y is involved in the largest number of constraints. As such, the backtracking search should pick variable y first.

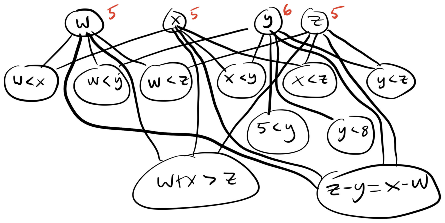


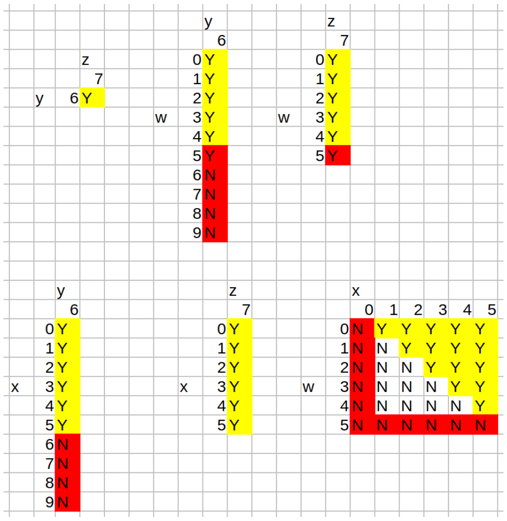
Figure 7: Constraint graph in which edges connect variable vertices to constraint vertices in which that variable occurs in the constraint. The red number indicates the degree of the nearby variable vertex.

(2) **From your answer in part 1, what value should be picked for this variable?**

The value which rules out the fewest choices for neighboring nodes in the constraint graph (i.e. other nodes which are connected to the same constraint vertex in figure 7) should be selected. This rules out values $0, 1, 2, 3, 4, 5, 8, 9$ since these values immediately result in a search failure as the constraint $5 < y < 8$ fails. That leaves values 6 and 7 as candidates. Looking at figure 6 from problem 4, we can see that for a value of y = 6, the domains of variables w, x, and z are reduced by 4, 5, and 7 values respectively (when only considering 2-consistent pairings involving y) totaling 16. For a value of y = 7, on the other hand, the domains of variables w, x, and z are reduced by 3, 4, and 8 values respectively (when only considering 2-consistent pairings involving y) totaling 15. Hence, a value of y = 6 should be chosen so as to rule out the fewest number of choices for neighboring nodes.

Continued on next page ⟶

(3) **Ignoring parts (1) and (2), assume you have assigned {z=7, y=6}. What are the domains of the remaining variables that are 2-consistent?**



		z
		7
y	6	Y

		y
		6
w	0	Y
	1	Y
	2	Y
	3	Y
	4	Y
	5	Y
	6	N
	7	N
	8	N
	9	N

		z
		7
w	0	Y
	1	Y
	2	Y
	3	Y
	4	Y
	5	Y

		y
		6
x	0	Y
	1	Y
	2	Y
	3	Y
	4	Y
	5	Y
	6	N
	7	N
	8	N
	9	N

		z
		7
x	0	Y
	1	Y
	2	Y
	3	Y
	4	Y
	5	Y

		x					
		0	1	2	3	4	5
w	0	N	Y	Y	Y	Y	Y
	1	N	N	Y	Y	Y	Y
	2	N	N	N	Y	Y	Y
	3	N	N	N	N	Y	Y
	4	N	N	N	N	N	Y
	5	N	N	N	N	N	N

Figure 8: Tables used to determine domains of 2-consistent variables

$\mathcal{D}_w = \{0, 1, 2, 3, 4\}$
$\mathcal{D}_x = \{1, 2, 3, 4, 5\}$
$\mathcal{D}_y = \{6\}$
$\mathcal{D}_z = \{7\}$

Problem 6. (10 points)

We will again test the N-queens problem (framed as a constraint satisfaction problem this time).

(1) **Report the run-time for board sizes = {11, 20, 28} when solving the N-Queens CSP problem with backtracking search.**

Board Size (N)	Average (n = 5) backtracking Search run-time
11	0.424 seconds
20	2.17 seconds
28	33.736 seconds

Table 1: Average backtracking search run-times (n = 5) for N-Queens CSP problem.

(2) **Report the run-time for board sizes = {11, 20, 28, 40} when solving the N-Queens CSP problem with min-conflicts.**

Board Size (N)	Average (n=5) Min-conflicts runtime
11	0.43 seconds
20	0.442 seconds
28	0.39 seconds
40	0.424 seconds

Table 2: Average run-times (n = 5) for N-Queens CSP problem solved using min-conflicts.

(3) **Find what board size takes approximately 10 seconds to solve with min-conflicts (make sure it is not ending early).**

With a board size of N = 4375, the average run-time over 5 samples was 9.966 seconds. The individual sample run-times were 10.02, 10.03, 9.88, 10.01, and 9.89 seconds.

Problem 6. (15 points)

(5/15 points) The classic problem is what is already put in as the Zebra problem. Run the `backtracking_search()` on this problem and report the answers to: Who owns the Zebra? And who drinks water? (It is fine to use the default parameters for `backtracking_search()`.

Backtracking search finds the following solution to the classic Zebra problem:

```
House 1 Yellow Fox Water Norwegian Kools
House 2 Blue Horse Tea Ukranian Chesterfields
House 3 Red Snails Milk Englishman Winston
House 4 Ivory Dog OJ Spaniard LuckyStrike
House 5 Green Zebra Coffee Japanese Parliaments
```

Therefore, the Japanese person in house 5 owns the Zebra, and the Norwegian person in house 1 drinks water.

(10/15 points) Modify the problem to match the Zebra problem below (with names, sports, transportation and lawns) and report: Who has flowers in their yard? Who likes to watch Starcraft2?

- **There are five houses.**
- **Albert likes baseball.**
- **Dietfried has a lawn with cleanly cut grass.**
- **Virgilijus enjoys watching rugby.**
- **Gallchobhar gets to work by walking.**
- **Bricius lives next to Virgilijus.**
- **The owner of the house with an Astroturf lawn likes baseball.**
- **The house with flowers in the lawn is between (directly next to) the house with trees and the house with rocks in the lawns.**
- **The person who likes baseball goes to work on a motorcycle.**
- **The 2nd house owners like to watch debates.**
- **The owner of the 4th house goes to work on a bicycle.**
- **The person who likes soccer takes a bus to work.**
- **The house with trees on the lawn to the right of the house with an owner who drives a car to work.**
- **Gallchobhar lives to the right of the person who likes Rugby.**

Backtracking search finds the following solution to the modified Zebra problem described above:

```
House 1 astroturf baseball Albert motorcycle
House 2 cleanly_cut_grass debates Dietfried car
House 3 trees soccer Bricius bus
House 4 flowers rugby Virgilijus bicycle
House 5 rocks starcraft2 Gallchobhar walking
```

Therefore, Virgilijus has flowers in their yard, and Gallchobhar likes to watch Starcraft2.

Continued on next page ⟶

Code for modified zebra problem:

```
def Zebra2():
    """Return an instance of the Zebra Puzzle."""
    lawn = 'cleanly_cut_grass astroturf flowers trees rocks'.split()
    sport = 'baseball rugby soccer debates starcraft2'.split()
    names = 'Albert Dietfried Virgilijus Gallchobhar Bricius'.split()
    transportation = 'walking motorcycle bicycle bus car'.split()
    variables = lawn + sport + names + transportation
    domains = {}
    for var in variables:
        domains[var] = list(range(1, 6))
    domains['debates'] = [2]
    domains['bicycle'] = [4]
    neighbors = parse_neighbors("""Albert: baseball;
                Dietfried: cleanly_cut_grass; Virgilijus: rugby; Gallchobhar: walking;
                Bricius: Virgilijus; astroturf: baseball; flowers: trees;
                flowers: rocks; baseball: motorcycle; soccer: bus;
                trees: car; Gallchobhar: rugby""")
    for type in [lawn, sport, names, transportation]:
        for A in type:
            for B in type:
                if A != B:
                    if B not in neighbors[A]:
                        neighbors[A].append(B)
                    if A not in neighbors[B]:
                        neighbors[B].append(A)

    def zebra_constraint_2(A, a, B, b, recurse=0):
        same = (a == b)
        next_to = abs(a - b) == 1
        if A == 'Albert' and B == 'baseball':
            return same
        if A == 'Dietfried' and B == 'cleanly_cut_grass':
            return same
        if A == 'Virgilijus' and B == 'rugby':
            return same
        if A == 'Gallchobhar' and B == 'walking':
            return same
        if A == 'Bricius' and B == 'Virgilijus':
            return next_to
        if A == 'astroturf' and B == 'baseball':
            return same
        if A == 'flowers' and B == 'trees':
            return next_to
        if A == 'flowers' and B == 'rocks':
            return next_to
        if A == 'baseball' and B == 'motorcycle':
            return same
        if A == 'soccer' and B == 'bus':
            return same
        if A == 'trees' and B == 'car':
            return a - 1 == b
        if A == 'Gallchobhar' and B == 'rugby':
            return a - 1 == b
        if recurse == 0:
            return zebra_constraint_2(B, b, A, a, 1)
        if ((A in lawn and B in lawn) or
              (A in sport and B in sport) or
              (A in names and B in names) or
              (A in transportation and B in transportation)):
            return not same
        raise Exception('error')

    return CSP(variables, domains, neighbors, zebra_constraint_2)


def solve_zebra2(algorithm=min_conflicts, **args):
    z = Zebra2()
    ans = algorithm(z, **args)
    for h in range(1, 6):
        print('House', h, end=' ')
        for (var, val) in ans.items():
            if val == h:
                print(var, end=' ')
        print()
    return ans['flowers'], ans['starcraft2'], z.nassigns, ans
```