1.1. The variables are $\{A, B, C, D, E\}$, representing the different classes. The domains are

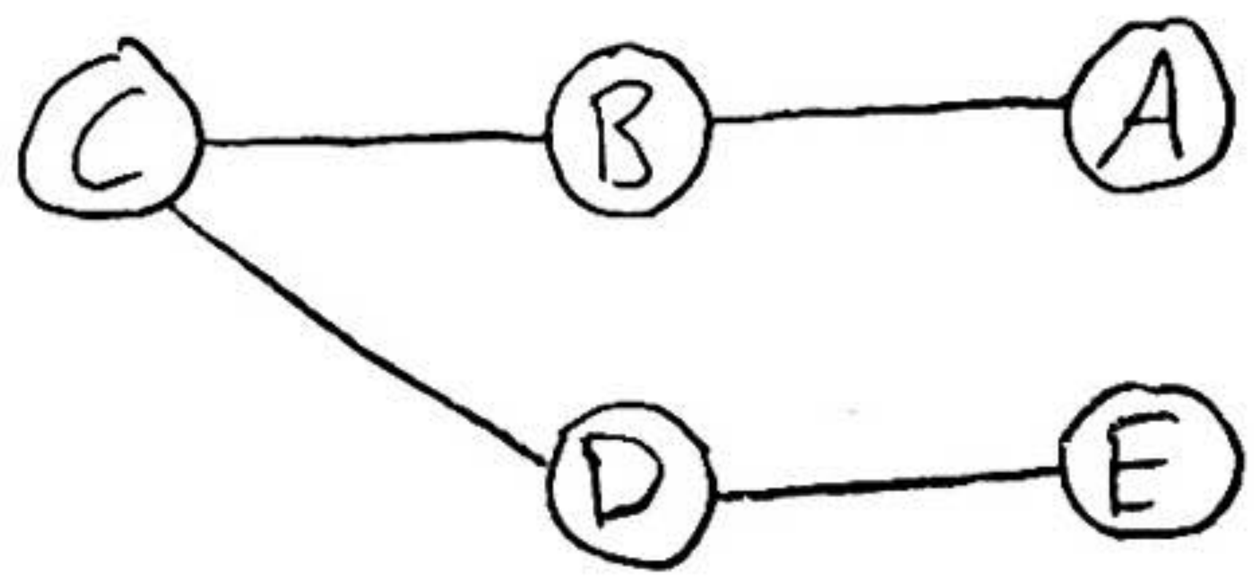
$D_A = \{Z\}$, $D_B = \{Y, Z\}$, $D_C = \{X, Y, Z\}$,

$D_D = \{X, Y, Z\}$, $D_E = \{Y, Z\}$, and the domain is $\{D_A, D_B, D_C, D_D, D_E\}$. The constraints

are $\{\langle (A, B), A \neq B \rangle, \langle (B, C), B \neq C \rangle,$

$\langle (C, D), C \neq D \rangle, \langle (D, E), D \neq E \rangle\}$.

.2.



.3. The new CSP after enforcing arc-consistency has the following domains:

$D_A = \{Z\}$, $D_B = \{Y\}$, $D_C = \{X, Z\}$, $D_D = \{X, Y, Z\}$, $D_E = \{Y, Z\}$.

This new CSP is easier to solve because any assignment to variables from their respective domains is now a valid one. A solution to the new arc-consistent CSP is $\{A = Z, B = Y, C = X, D = Y, E = Z\}$, and this assignment is also a solution to the original CSP.

2.1. We can model a standard 9×9 Sudoku game as a CSP by making the variables the squares in the grid (e.g., $\{S_{ij}\}$ where (i,j) is the coordinate of the square), the domains all $D_{ij} = \{1,2,3,4,5,6,7,8,9\}$, and the constraints $\text{Alldiff}(D_{ij})$ for each set of (i,j)'s in the same column, each set of (i,j)'s in the same row, and each set of (i,j)'s in the same one of the 9 3×3 subgrids, as well as the unary constraints $\langle (S_{ij}), S_{ij} = N_{ij} \rangle$ for each N_{ij} representing already filled-in squares. The constraints are a mix of unary and global constraints. I suppose that the 'Alldiff' constraints can be decomposed into binary constraints by having $\langle (S_1, S_2), S_1 \neq S_2 \rangle$ for each distinct pair of variables S_1, S_2 in the input set to 'Alldiff'.
Backtracking search differs from brute-force guess-and-check in that we consider <u>commutativity</u>.

2.2. See pset2.py

2.3. See pset2.py

2.4. The average amount of backtracking search required to find a solution INCREASES as the number of values on the starting board DECREASES. This is because the inclusion of set values for squares can be thought of as a pre-forward-checking; work that has <u>already</u> been done for us. In the case where there is very few values provided, such as just one, backtracking search must enforce arc-consistencies more times per square that is empty. In other words, the more solutions available, the longer it will take to find one.

2.5. My heuristic picks the candidate with <u>least frequency</u> among the <u>candidates of related coordinates when known</u> (e.g., when in Gameboard.coordinate_dict. The justification is that the solution will contain an even distribution of the integers and so it makes sense to pick one not yet seen as often from the candidates of related coordinates.

3.1. a) $f: \mathbb{R}^n \to \mathbb{R}, \ f(\vec{x}) = \max_i x_i$

Proof Let $\vec{x}, \vec{y} \in \mathbb{R}^n$. Then, for any $\theta \in [0,1]$,

$$f(\theta\vec{x} + (1-\theta)\vec{y}) = \max_i(\theta x_i + (1-\theta)y_i)$$

$$= \theta \max_i x_i + (1-\theta)\max_i y_i$$

$$= \theta f(\vec{x}) + (1-\theta) f(\vec{y})$$

$$\Rightarrow f(\theta\vec{x} + (1-\theta)\vec{y}) \leq \theta f(\vec{x}) + (1-\theta) f(\vec{y})$$

which means f is convex. ■

3.1. b) $f: \mathbb{R}^2_+ \to \mathbb{R}, \ f((x,y)) = x/y, \quad \mathbb{R}^2_+ := \{(x,y) \in \mathbb{R}^2 \mid y > 0\}$

Proof. Let $(x_1,y_1), (x_2,y_2) \in \mathbb{R}^2_+$. Then, for any $\theta \in [0,1]$,

$$f(\theta(x_1,y_1) + (1-\theta)(x_2,y_2)) = f((\theta x_1 + (1-\theta)x_2, \theta y_1 + (1-\theta)y_2))$$

$$= \frac{\theta x_1 + (1-\theta)x_2}{\theta y_1 + (1-\theta)y_2}$$

$$= \frac{\theta x_1}{\theta y_1 + (1-\theta y_2)} + \frac{(1-\theta)x_2}{\theta y_1 + (1-\theta)y_2}$$

$$< \frac{\theta x_1}{\theta y_1} + \frac{(1-\theta)x_2}{(1-\theta)y_2}$$

$$= \theta f((x_1,y_1)) + (1-\theta) f((x_2,y_2))$$

$$\Rightarrow f(\theta(x_1,y_1) + (1-\theta)(x_2,y_2)) \leq \theta f((x_1,y_1)) + (1-\theta) f((x_2,y_2))$$

which means f is convex. ■

3.1. c) $f: \mathbb{R} \to \mathbb{R}$. $f(x) = \max\{f_1(x), \dots, f_n(x)\}$ where $f_1, \dots, f_n$ are convex functions $\mathbb{R} \to \mathbb{R}$

Proof. Let $x, y \in \mathbb{R}$. For any $\theta \in [0,1]$,

1: $f(\theta x + (1-\theta)y) = \max\{f_1(\theta x + (1-\theta)y), \dots, f_n(\theta x + (1-\theta)y)\}$

2: Since $f_1, \dots, f_n$ are convex, we have that for $i \in 1, \dots, n$,

$$f_i(\theta x + (1-\theta)y) \le \theta f_i(x) + (1-\theta) f_i(y).$$

3: 1 and 2 imply that

$$\max\{f_1(\theta x + (1-\theta)y), \dots, f_n(\theta x + (1-\theta)y)\}$$

$$\le \max\{\theta f_1(x) + (1-\theta) f_1(y), \dots, \theta f_n(x) + (1-\theta) f_n(y)\}$$

and it's important to note that the function f_i corresponding to the largest term in the left-hand 'max' of the inequality will NOT necessarily correspond to the largest term in the right-hand 'max'. The inequality holds nonetheless.

4: 3 and 1 imply $f(\theta x + (1-\theta)y)$

$$\le \max\{\theta f_1(x) + (1-\theta) f_1(y), \dots, \theta f_n(x) + (1-\theta) f_n(y)\}$$

$$= \theta \max\{f_1(x), \dots, f_n(x)\} + (1-\theta)\max\{f_1(y), \dots, f_n(y)\}$$

$$= \theta f(x) + (1-\theta) f(y)$$

which means f is convex. ∎

3.2. a)

Let $C \subseteq \mathbb{R}^n$ be a convex set with $x_1, \dots, x_n \in C$, and let $\theta_1, \dots, \theta_k \in \mathbb{R}$ satisfy $\theta_i \geq 0$, $\theta_1 + \dots + \theta_k = 1$ for any $k \geq 1$.

<u>Claim</u>: $\theta_1 x_1 + \dots + \theta_k x_k \in C$.

<u>Proof by induction</u>:

Base cases: $k=1$: Then $\theta_1 = 1$ and $\theta_1 x_1 = x_1$ which is in C. $k=2$: Then $\theta_2 = (1-\theta_1)$ and $\theta_1 x_1 + (1-\theta_1) x_2 \in C$ holds since C is convex.

Inductive step: We show that if the claim holds for $k=m$, it holds for $k=m+1$.

Proof. Assume the claim holds for $k=m$. then

$\theta_1 x_1 + \dots + \theta_m x_m \in C$. Let $x_{m+1} \in C$.

Then, since C is convex,

$$\theta'(\theta_1 x_1 + \dots + \theta_m x_m) + (1-\theta') x_{m+1} \in C$$

$$\Rightarrow \theta'\theta_1 x + \dots + \theta'\theta_m x_m + (1-\theta') x_{m+1} \in C$$

Notice that we can rename $\theta'\theta_1, \dots, \theta'\theta_m, (1-\theta')$ as $\theta_1^{new}, \dots \theta_{m+1}^{new}$, and that there new variables (next page)

3.2. a) (cont.)

can be any that satisfy $\theta_i^{new} \geq 0$, $\theta_1^{new} + \ldots + \theta_{m+1}^{new} = 1$, since the original $\theta_1, \ldots, \theta_m$ can be tailored to produce any such set. This means that

$$\theta_1^{new} x_1 + \ldots + \theta_{m+1}^{new} x_{m+1} \in C$$

for any $\theta_1^{new} \ldots \theta_{m+1}^{new}$ that satisfy $\theta_i^{new} \geq 0$, and $\theta_1^{new} + \ldots + \theta_{m+1}^{new} = 1$. This is what we wanted to show in our inductive step, that if the claim holds for $k=m$, it holds for $k=m+1$.

$\Rightarrow$ We have shown that the claim holds for $k=1$, $k=2$, and that if the claim holds for $k=m$ it holds for $k=m+1$. Thus, the claim holds for $k \geq 1$ ■

3.2. b) i)

Proof. Let $a, b \in S$. Then, for any $\theta \in [0,1]$

① $\theta a + (1-\theta) b = \theta(\theta_1^{(1)} a_1 + \ldots + \theta_k^{(1)} a_k)$
$+ (1-\theta)(\theta_1^{(2)} b_1 + \ldots + \theta_\ell^{(2)} b_\ell)$

for some $\theta_1^{(1)}, \ldots, \theta_k^{(1)}$ and $\theta_1^{(2)}, \ldots, \theta_\ell^{(2)}$ satisfying $\sum_i \theta_i^{(1)} = 1$ and $\sum_i \theta_i^{(2)} = 1$, $a_1, \ldots, a_k, b_1, \ldots, b_\ell \in C$.

3.2.b) i) (cont.)

Eq. (3) implies

$$\theta a + (1-\theta) b = \theta\theta_1^{(1)} a_1 + \dots + \theta\theta_k^{(1)} a_k$$
$$+ (1-\theta)\theta_1^{(2)} b_1 + \dots + (1-\theta)\theta_\ell^{(2)} b_\ell$$

We have that

$$\theta\theta_1^{(1)} + \dots + \theta\theta_k^{(1)} + (1-\theta)\theta_1^{(2)} + \dots + (1-\theta)\theta_\ell^{(2)} = 1$$

since $\theta + (1-\theta) = 1$ and

$$\theta_1^{(1)} + \dots + \theta_k^{(1)} = \theta_1^{(2)} + \dots + \theta_\ell^{(2)} = 1.$$

We also have that $a_1, \dots, a_k, b_1, \dots, b_\ell \in C$.

Hence, by the definition of C,

$\theta a + (1-\theta) b \in C$ and this means

C is convex. ∎

3.2.b) ii)

Proof. Claim 1: A convex set containing S contains $CH(S)$.

Proof: This is shown in 3.2.c) i): if

$x_1, \dots, x_k \in C$, then $\theta_1 x_1, \dots, \theta_k x_k \in C$ for

any $\theta_1, \dots, \theta_k$ satisfying $\theta_i > 0$, $\sum_i \theta_i = 1$, for all $k \geq 1$.

Claim 2: $CH(S)$ is a set containing S.

Proof: S itself is a convex combination of the points in S.

Claim 1 and 2 imply that the intersection of all convex sets that contain S is $CH(S)$, →

3.2. b) ii) (cont.) since the intersection of all such sets will contain CH(s) (claim 1), and one of these will ONLY contain CH(s) (claim 2). ∎

4. 1.

		Paul R	Paul G	Paul B
Fiona	R	7, 1	$\underline{9}$, 2	6, $\underline{3}$
	G	$\underline{8}$, $\underline{10}$	7, 5	7, 1
	B	7, $\underline{2}$	8, $\underline{2}$	$\underline{8}$, -3

- A pure Nash equilibrium is $\boxed{\text{Fiona: green, Paul: red}}$ since at this point (F = 8, P = 10), neither Fiona or Paul can do any better by changing their choice of color.
- Let p, q, 1-p-q be the probabilities Fiona plays red, green, blue, respectively. Likewise, let x, y, 1-x-y be the corresponding probabilities for Paul.

For Fiona to be indifferent between red and green, we need ① $7x + 9y + 6(1-x-y) = 8x + 7y + 7(1-x-y)$

likewise, for Fiona to be indifferent b/w green and blue, we need ② $8x + 7y + 7(1-x-y) = 7x + 8y + 8(1-x-y)$.

↳

4.1. (cont.)

Likewise, the equations that determine Paul's indifference to red, green, blue are

$$\textcircled{3}\quad p + 10q + 2(1-p-q) = 2p + 5q + 2(1-p-q)$$

$$\textcircled{4}\quad 2p + 5q + 2(1-p-q) = 3p + q - 3(1-p-q)$$

Solving ①, ②, ③, ④ yields only one solution with strictly positive probabilities:

$$p = 25/31,\ q = 5/31,\ x = 1/2,\ y = 1/3$$

$$\Downarrow$$

the mixed Nash equilibrium is

$$\boxed{\begin{array}{l} \text{Fiona: red} = 25/31,\ \text{green} = 5/31,\ \text{blue} = 1/31 \\ \text{Paul: red} = 1/2,\ \text{green} = 1/3,\ \text{blue} = 1/6 \end{array}}$$

4.2. a) There is no pure Nash equilibrium. This is because: ① if $x_1 = x_2 = x_3$, all are motivated to move anywhere else else they are currently getting $1/3$ of the customers and there is $> 1/3$ of the candidates on one of the sides. ② if $x_1 = x_2 \neq x_3$, x_3 gets more customers by moving closer to x_1. (same for $x_1 \neq x_2 = x_3$ or $x_1 = x_3 \neq x_1$).

4.2.a) (cont.)

3 if $x_1 < x_2 < x_3$, either x_1 or x_3 can get more customers by moving closer to x_2. Same for any other order (e.g. $x_1 < x_3 < 2$).

1, 2, 3 are collectively exhaustive of all possibilities and so there are no pure NE.

4.2.b) $\boxed{\text{There is a pure NE}}$ when two of the sellers are at $\boxed{1/4}$ and the other two are at $\boxed{3/4}$. This is a pure NE since all players are receiving 1/4 of the customers and there are only 1/4 candidates available for them to "steal" if they move anywhere (no benefit).

5.1. See pset2.py

5.2.a) The variables are the entries of the matrix of size $N \times K$ denoted $x_{n,k}$.

The constraints are that

1 $x_{n,k} \in [0,1]$, $\forall n,k$,

2 $\sum_i x_{n,i} = 1 \quad \forall n$.

5.2.a) (cont.)

Constraint ① guarantees each $x_{n,k}$ will represent a boolean expressing whether task n was assigned to machine k, and ② guarantees each task is assigned to only one machine.

The objective is to <u>minimize</u>

$$\max\{\text{WORKTIMES}\}$$

where WORKTIMES is $\left[\left(\sum_i x_{i,1} \cdot p_i\right), \ldots, \left(\sum_i x_{i,n} \cdot p_i\right)\right]$

where each term $\left(\sum_i x_{i,j} \cdot p_i\right)$ corresponds to the time it takes machine j to finish its assigned tasks.

5.2.b. See pset2.py. A lower bound on the objective function is $\max_i p_1, \ldots, p_N$ since the makespan can never be lower than the longest individual task. Another comes from the LP relaxation of the problem: The mean of $p_1, \ldots, p_N$ divided by K.

6. 1) No one. None

2) >20 hours. Way too long IMO even for a CS class.