CIS 510 Assignment 2

Steven Walton

May 17, 2019

Problem P1

Program expects an input where the first line is the number of players and each subsequent line is the grand coalition of a subset of players surrounded by curly braces and comma separated with the value of that coalition. An example input file is shown here

```
4
{1},30
{2},40
{3},25
{4},45
{1,2},50
{1,3},60
{1,4},80
{2,3},55
{2,4},70
{3,4},80
{1,2,3},90
{1,2,4},120
{1,3,4},100
{2,3,4},115
{1,2,3,4},140
```

The program can be run with the following options. Not that only the input file is required, but it is suggested to specify the output file.

```
python coalition.py --help
usage: coalition.py -i <input file> -o <output file>
-h, --help          prints this message
-i, --input         sets the input file
-o, --output        sets the output file: defaults to optimalCS.txt
```

Problem Q1

Consider the “cross-out” game. In this game one writes down ”1,2,3”. Player 1 can cross out a single number or any 2 adjacent number (12,23). Player 2 then gets to make the same type of action. The winner is the one who crosses out the last number.

Part Q1.1)

We'll create a tree where the text within the nodes denote what available choices to the current player and where the edges show what numbers the player picked.

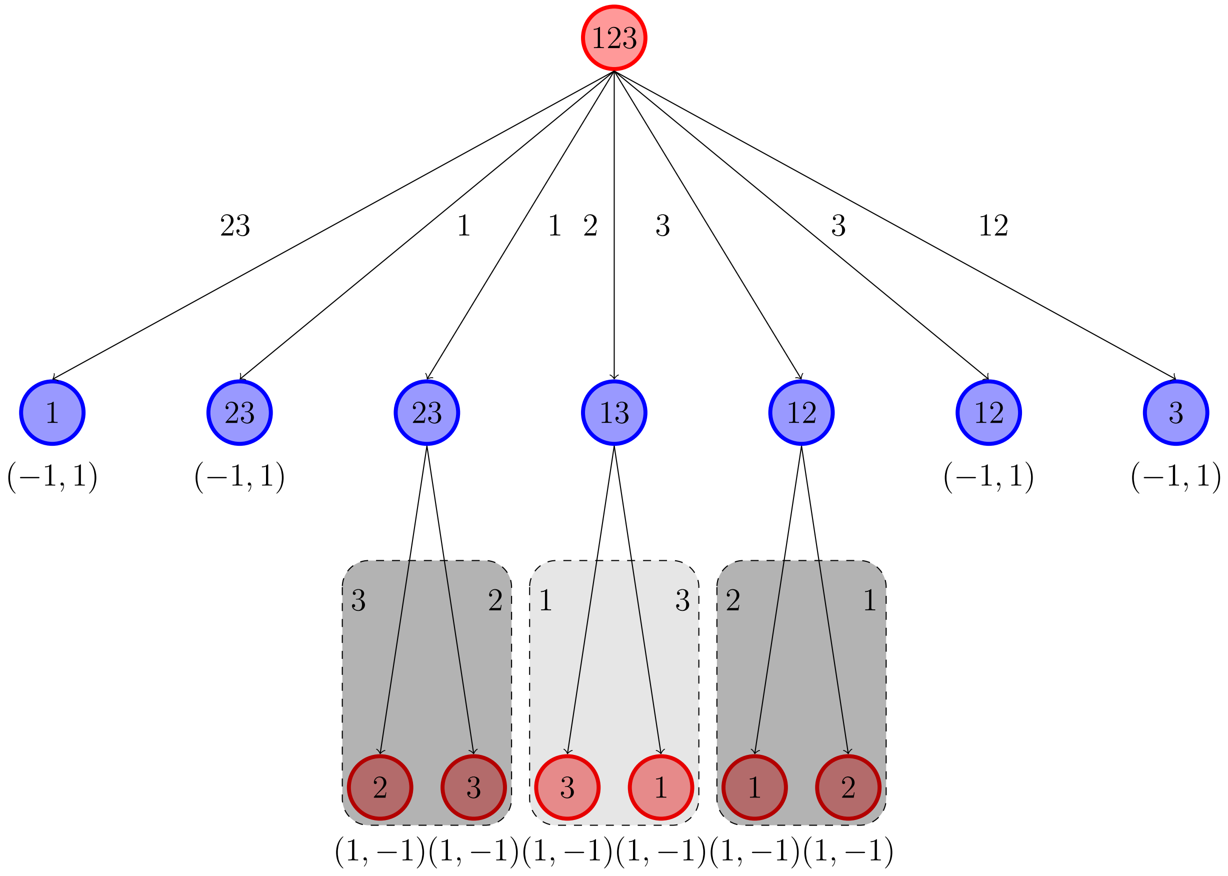

Figure 1: Nim game: Pick a maximum of 2 adjacent numbers

Looking at Figure 1 we can see that the boxed subgames player 1 can always win if player 2 picks a single number. The darker boxes denote where player 2 can win by crossing out adjacent numbers (this is equivalent to choosing the blue nodes adjacent to these boxes and player 2 finishing the game).

This leaves two sub perfect games where player 1 always wins. Player one picks either {2,3} or {2,1} and player 2 can choose either {1} or {3} (respective to player 1's plays). Player 1 dominates this game because they can force player 2 into a losing action.

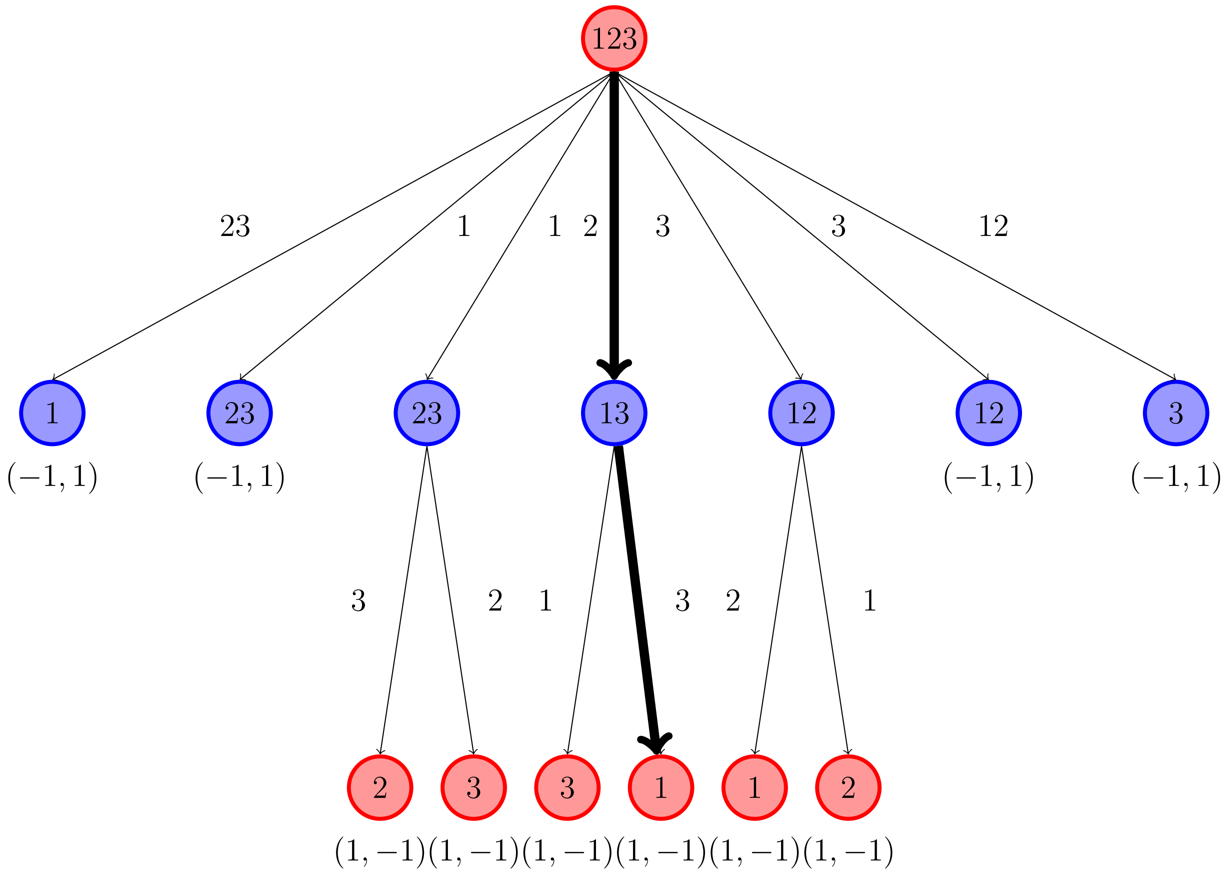

Figure 2: Nim game: Subgame Perfect Nash Equilibrium 2-3-1

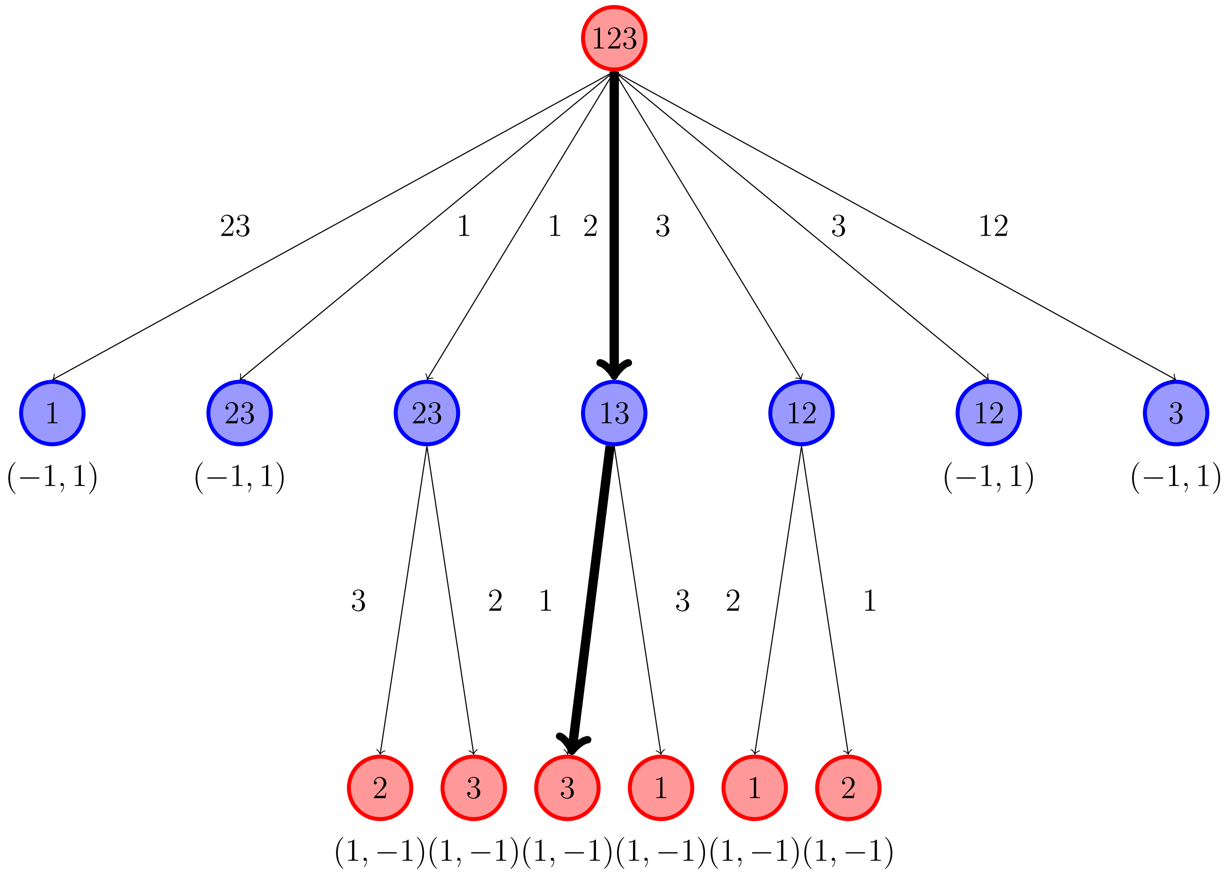

Figure 3: Nim game: Subgame Perfect Nash Equilibrium 2-1-3

Part Q1.2)

This is a solved game where player 1 can always win.
First let's consider the game where a player could only cross out a single number. This game is also solved! If it is an odd sized game then player 1 can always win, otherwise player 2 can always win. This is because if the size is odd then player 1 can place a binary partition and split the game into any two sub games (like how in Figure 1 Player 1 splits the game into two odd sized games by picking 2). Player 2 will then be the first player on one sub game and player 1 will be the second player in the second sub game (assuming players play optimally, this is easy to see). If the second player wins any even game, then we can see that player 1 will always win this sub game (because it is even and they are the second player for an even sized game). This tells us how to win many different variations of this game! The key is to be a specific player and control if the game is even or odd. By being able to pick up to two numbers, Player 1 can always make this control.
As an example, let's assume that Player 1 wants to keep a subgame odd. If Player 2 picks 1 then Player 1 picks 2 (because $1 + 2 = 3$ which is odd). If Player 2 picks 2, then Player 1 picks 1 (because $2 + 1 = 3$ which is odd). Similarly, Player 1 can keep

	L	R	Ll	Lr	LlU	LlD
A	0	(1,1)	0	(2,3)	0	0
B	0	(1,1)	0	(3,2)	0	0
AC	0	(1,1)	0	(2,3)	(4,1)	(1,4)
AD	0	(1,1)	(3,4)	(2,3)	(3,4)	(3,4)
BE	0	(1,1)	(2,5)	(3,2)	(2,5)	(2,5)
BF	0	(1,1)	0	(3,2)	(2,3)	(3,2)

Table 1: Payoff Structure of Game

a subgame even.
Knowing this, all Player 1 needs to do is keep track of the subgames and which ones they need to win or lose (all that matters is that they win the last one).

Problem 2

Part Q2.1)

Realization Plan Player 1

$$\begin{aligned}
r_1(\oslash) &= r_1(L) + r_1(R) = 1 \\
r_1(L) &= r_1(Ll) + r_1(Lr) = 0.6 \\
r_1(R) &= r_1(Rl) + r_1(Rl) = 0.4 \\
r_1(Ll) &= r_1(LlU) + r_1(LlD) = 0.4 \\
r_1(Lr) &= r_1(LrU) + r_1(LrD) = 0.2 \\
r_1(LlU) &= 0.2 \\
r_1(LlD) &= 0.2 \\
r_1(LrU) &= 0.1 \\
r_1(LrD) &= 0.1 \\
r_1(\oslash), r_1(L), r_1(R)&, r_1(Ll), r_1(Lr), r_1(LlU), r_1(LlD), r_1(LrU), r_1(LrD) \geq 0
\end{aligned}$$

Realization Plan Player 2

$$
\begin{aligned}
r_2(\oslash) &= r_2(A) + r_2(B) = 1 \\
r_2(A) &= r_2(AC) + r_2(AD) = 0.5 \\
r_2(B) &= r_2(BC) + r_2(BD) = 0.5 \\
r_2(AC) &= 0 \\
r_2(AD) &= 0.5 \\
r_2(BE) &= 0.5 \\
r_2(BF) &= 0 \\
r_2(\oslash), r_2(A), r_2(B), &r_2(AC), r_2(AD), r_2(BE), r_2(BF) \geq 0
\end{aligned}
$$

Strategies AC and BF are set to 0 because under those circumstances it would never be a good idea for Player 2 to take those actions.

Part Q2.2)

We have
Player 1:

$$
\begin{aligned}
\beta_1(L) &= r_1(L) = \frac{3}{5} & \beta_1(R) &= r_(R) = \frac{2}{5} \\
\beta_1(l) &= \frac{r_1(Ll)}{r_1(L)} = \frac{2}{3} & \beta_1(r) &= \frac{r_1(Lr)}{r_1(L)} = \frac{1}{3} \\
\beta_1(U) &= \frac{r_1(LlU)}{r_1(Ll)} = \frac{1}{2} & \beta_1(D) &= \frac{r_1(LlD)}{r_1(Ll)} = \frac{1}{2}
\end{aligned}
$$

Player 2:

$$
\begin{aligned}
\beta_2(A) &= r_2(A) = \frac{1}{2} & \beta_2(B) &= r_2(B) = \frac{1}{2} \\
\beta_2(C) &= \frac{r_2(AC)}{r_2(A)} = 0 & \beta_2(D) &= \frac{r_2(AD)}{r_2(A)} = 1 \\
\beta_2(E) &= \frac{r_2(BE)}{r_2(B)} = 1 & \beta_2(F) &= \frac{r_2(BF)}{r_2(B)} = 0
\end{aligned}
$$

Part Q2.3)

$$
\begin{aligned}
&\max \sum_{\sigma_2 \in \Sigma_2} \left(\sum_{\sigma_1 \in \Sigma_1} g_2(\sigma_1, \sigma_2) r_1(\sigma_2) \right) r_2(\sigma_2) \\
&s.t \quad r_2(\oslash) = 1 \\
&\qquad \sum_{\sigma_2' \in Ext_2(I)} r_2(\sigma_2') = r_2(seq_2(I)) \\
&\qquad r_2(\sigma_2) \geq 0
\end{aligned}
$$

Where $g_2(\sigma_1, \sigma_2)$ are the combinations or realization plans from player 1 and 2, where $\sigma_1 \in \Sigma_1$ and $\sigma_2 \in \Sigma_2$

Σ_1	Σ_2
$\oslash$	$\oslash$
L	A
R	B
Ll	AC
Lr	AD
LlU	BE
LlD	BF

As a mathematician I thought we were done here, because this is a compact representation of all the information needed to solve the problem and this could could programmatically be done.
We know that g_2 and r_1 are constants and that r_2 is the only variable. We can do some careful reduction to make the expansion easier. We note that all $g(*, \oslash) = 0$ Similarly any plan that does not reach a leaf node results in 0. We use $*$ similar to a wild character.

$$\begin{aligned}
&\max \sum_{\sigma_2 \in \Sigma_2} r_2(\sigma_2) \left(\sum_{\sigma_1 \in \Sigma_1} g_2(\sigma_1, \sigma_2) r_1(\sigma_1) \right) \\
&= \max r_2(\oslash)(0) \\
&+ r_2(A)[g(R, A)r_1(R) + g(Lr, A)r_1(Lr)] \\
&+ r_2(AD)[g(R, AD)r_1(R) + g(Lr, AD)r_1(Lr) + g(Ll, AD)r_1(Ll)] \\
&+ r_2(B)[g(R, B)r_1(R) + g(Lr, B)r_1(Lr)] \\
&+ r_2(BE)[g(R, BE)r_1(R) + g(Lr, BE)r_1(Lr) + g(Ll, BE)r_1(Ll)] \\
&= \max r_2(A)[(1)(0.4) + (3)(0.2)] + r_2(AD)[(1)(0.4) + (3)(0.2) + (4)(0.2)] \\
&+ r_2(B)[(1)(0.4) + (2)(0.4)] + r_2(BE)[(1)(0.4) + (2)(0.2) + (5)(0.4)] \\
&= \max (r_2(A) + r_2(AD)(1.8) + r_2(B)(1.2) + r_2(BE)(2.8))
\end{aligned}$$