Contents

Summary

We are exploring the popular board game 'The Resistance' through a formal logic lens, where the turn-based format with 2 teams (Spies, Resistance) and hidden player identities allow the establishment of propositional models that are solvable by computer. The 7 players go on a series of 'missions' which are voted to go ahead or not by a subset of all players selected by a leader in each of the 5 rounds. A mission succeeds for the Resistance team and gains them a point if all participants submit a 'success' card in an anonymous choice procedure, which is the key game mechanic as a given player doesn't directly know the team membership of other players and is forced to infer this information as the rounds progress. Spies can submit a 'fail' card and cause the mission to fail which gives them one point, and either team wins the game with 3 points.

Implementation proved to be non-trivial due to assigning models encompassing all possible play combinations per round and evaluating their satisfiability. Critically, votes were assigned randomly to simulate player behaviour which enabled a method of calculating the likelihood of the Resistance team winning at each stage of the game, as suggested by Prof. Muise after the proposal to properly expand the scope of our project.

Propositions

X_r is true at position r depending on the round number (e.g. X_2 is true for r2)
Z_s is true if current mission is rejected s times (e.g. Z_3 means the 3rd consecutive rejection)
Note that r and s both range from 1 to 5, but X_r and Z_s don't necessarily have the same truth value when $r = s$ due to how the game functions
Y_k is true if mission k is approved, false if not (e.g. $\neg Y_4$ means mission 4 not approved)
P_i is true if player i has voted to approve the current mission and false otherwise, where i ranges from 1 to 6
Q_j is true if player j has played a success token in the currently approved mission, false if player j has played a fail token (e.g. $\neg Q_1$ means player 1 has played a fail token)
Note that i ranges from either 1 to 3 or 1 to 4 depending on the number of

players per round
S_r is true if the current mission is a success for team Resistance and false otherwise, and vice-versa for team Spies
V represents the overall game victory condition: true for the Resistance winning the game, false for the Spies emerging victorious

Constraints

Detailed description of the game rules:
7 players total → 4 R, 3 S
5 total rounds ('missions')→ a team wins if they succeed in 3/5 in any order

Round structure:
r1: 3 players
r2: 4 players
r3: 3 players
r4: 4 players
r5: 4 players

Before the game starts (i.e. r0) each player is randomly assigned a team which none of the other players know. This assignment lasts until the game ends. At the start of each round, one player is assigned the role of 'mission leader' & the remaining 6 are given 1 accept and 1 reject token for voting purposes. The leader then gets to choose the corresponding number of players per round from the pool to send on the mission as above (leader not included).

Mission start:

Selected players vote to either accept or reject the current mission.
If a majority reject the mission, then the next counterclockwise player is assigned leader. NOTE: If 5 rejections occur in a row then the S team wins automatically. The vote tracker resets every round (e.g. r2 can have 1 rejection, r3 0, r4 3, etc.)
If a majority accept the mission, then each active player is given 1 success card and 1 failure card. NOTE: R players have to play success every time, whereas S players can play either success or failure. Once all players turn in their cards then the total number is counted. If there's $\geq$1 fail cards then the current mission fails → 1 point for team S. Otherwise, 1 point for team R

If there's a tie of accept/reject tokens (possible on rounds w/ an even number of players) then flip a coin to see whether the mission happens or not (H yes, T no).

Mission end

At the end of any round, if a team gets 3 points then they are declared the winner, unless the above win condition for team S is satisfied.

As per Prof. Moose's suggestion, we've expanded the scope of the constraints by imagining that at the time of player selection for each mission the leader will have access to a computer with the Python scripts for this project, and they will attempt to calculate an estimate of winning based on the probability that the other players are on team R or team S. This is done by simply counting the number of valid models (propositional formulas evaluating to true under the selected conditions) and dividing by the total number of models:

$$P(win) = \frac{n_{valid}}{n_{valid} + n_{invalid}} \tag{0.1}$$

We proceed through an example game with specific votes & tokens to illustrate how the constraints are expressed in propositional logic.

Round 1:

$X_1 \wedge (\neg P_1 \wedge \neg P_2 \wedge P_3 \wedge \neg P_4 \wedge \neg P_5 \wedge P_6) \rightarrow \neg Y_1 \wedge Z_1$

Here 4/6 players voted to reject the mission so Y_1, Z_1 are updated.

$X_1 \wedge (P_1 \wedge \neg P_2 \wedge \neg P_3 \wedge \neg P_4 \wedge \neg P_5 \wedge \neg P_6) \rightarrow \neg Y_1 \wedge Z_1 \wedge Z_2$

Here 5/6 players voted to reject the mission, with Z_2 added accordingly.
...
Voting rounds 3 and 4 also result in rejections.
...

$X_1 \wedge (\neg P_1 \wedge \neg P_2 \wedge \neg P_3 \wedge \neg P_4 \wedge \neg P_5 \wedge \neg P_6) \rightarrow \neg Y_1 \wedge Z_1 \wedge Z_2 \wedge Z_3 \wedge Z_4 \wedge Z_5$
$Z_5 \rightarrow \neg V$

After 5 rejections, the Spies automatically win the game.

Alternatively, let's say the 1st mission is approved:
$X_1 \wedge (P_1 \wedge \neg P_2 \wedge P_3 \wedge P_4 \wedge \neg P_5 \wedge P_6) \rightarrow Y_1$

Only 2/6 players voted against so now the mission can go ahead.

$Y_1 \wedge (Q_1 \wedge Q_2 \wedge Q_3) \rightarrow S_1$

In this scenario round 1 counts as a success for the Resistance as none of the players put forward a fail token.

Round 2:
$X_2 \wedge (P_1 \wedge P_2 \wedge P_3 \wedge \neg P_4 \wedge P_5 \wedge P_6) \rightarrow Y_2$

$Y_2 \wedge (\neg Q_1 \wedge Q_2 \wedge Q_3 \wedge \neg Q_4) \rightarrow \neg S_1$

Here the mission is approved right away, and Spies win the round by playing a single failure token.

Round 3:
$X_3 \wedge (\neg P_1 \wedge \neg P_2 \wedge P_3 \wedge \neg P_4 \wedge P_5 \wedge \neg P_6) \rightarrow \neg Y_3 \wedge Z_1$

$X_3 \wedge (P_1 \wedge P_2 \wedge P_3 \wedge \neg P_4 \wedge P_5 \wedge P_6) \rightarrow Y_3 \wedge Z_1 \wedge \neg Z_2$

Here the mission is approved on the 2nd voting round; notice how the Z_s counter resets to 1 at every round.

$Y_3 \wedge (\neg Q_1 \wedge Q_2 \wedge Q_3) \rightarrow S_3$

The Q_j's and P_i's are also different per round. Here the Resistance wins mission 3.

...
Round 4 is a win for team Spies
...

Round 5:

$X_5 \wedge (P_1 \wedge P_2 \wedge P_3 \wedge P_4 \wedge P_5 \wedge P_6) \rightarrow Y_5$

$Y_5 \wedge (Q_1 \wedge Q_2 \wedge Q_3 \wedge Q_4) \rightarrow \neg S_5$

The final tally is then:

$S_1 \wedge \neg S_2 \wedge S_3 \wedge \neg S_4 \wedge S_5 \rightarrow V$

RESISTANCE WINS - YAHOO!

Alternatively, let's assume that the Spies win in rounds 1, 2, and 4:

$\neg S_1 \wedge \neg S_2 \wedge S_3 \wedge \neg S_4 \rightarrow \neg V$

In this case the Spies automatically win after round 4 finishes as they have the required 3/5 missions. It's also possible for the Resistance to win after having played only 3 rounds, etc.

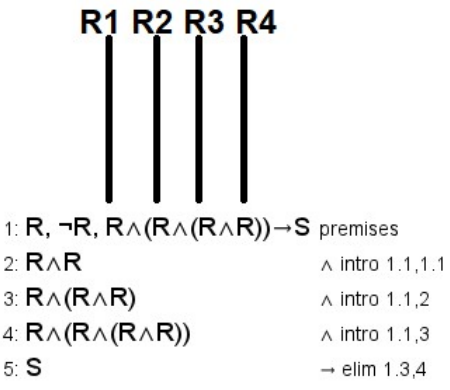
It's also possible to chain together a massive series of disjunctions for all the winning combinations of S_r's for the Resistance:

$$(S_1 \wedge S_2 \wedge S_3 \wedge \neg S_4 \wedge S_5) \vee (\neg S_1 \wedge S_2 \wedge S_3 \wedge S_4 \wedge S_5) \vee ... \vee (S_1 \wedge S_2 \wedge \neg S_3 \wedge \neg S_4 \wedge S_5) \rightarrow V \tag{0.2}$$

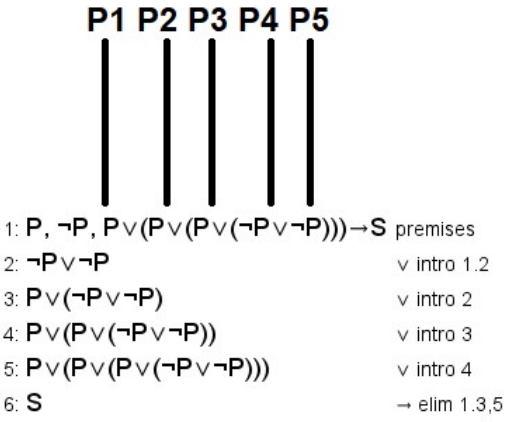
In the same way we can list all the possibilities that lead to a victory for the Spies. Bada-bing!

Jape Proofs

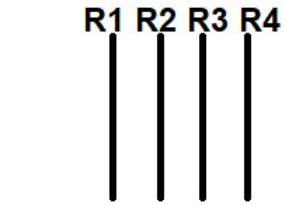
Please see the attached file proofs.jp.
We had trouble assigning subscripts, so the following images are for clarification purposes (variable names are different than in the Propositions section, i.e. $R_i = Q_j, S = S_r, Q = V$):



Ri represents the voting tracker during a mission.

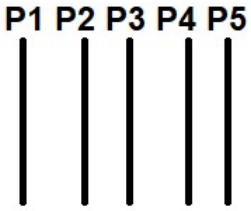


Pi represents the voting tracker after 5 completed rounds.



1:	$R, \neg R, R \wedge (R \wedge (R \wedge \neg R)) \rightarrow \neg S$	premises
2:	$R \wedge \neg R$	$\wedge$ intro 1.1,1.2
3:	$R \wedge (R \wedge \neg R)$	$\wedge$ intro 1.1,2
4:	$R \wedge (R \wedge (R \wedge \neg R))$	$\wedge$ intro 1.1,3
5:	$\neg S$	$\rightarrow$ elim 1.3,4

Ri represents the voting tracker during a mission.



1:	$P, \neg P, \neg P \vee (\neg P \vee (P \vee (\neg P \vee P))) \rightarrow \neg S$	premises
2:	$\neg P \vee P$	$\vee$ intro 1.1
3:	$P \vee (\neg P \vee P)$	$\vee$ intro 2
4:	$\neg P \vee (P \vee (\neg P \vee P))$	$\vee$ intro 3
5:	$\neg P \vee (\neg P \vee (P \vee (\neg P \vee P)))$	$\vee$ intro 4
6:	$\neg S$	$\rightarrow$ elim 1.3,5

Pi represents the voting tracker after 5 completed rounds.

1:	Q	premise
2:	$\boxed{\neg S}$	assumption
3:	$\boxed{Q}$	hyp 1
4:	$\neg S \rightarrow Q$	$\rightarrow$ intro 2-3

Q represents a victory. ~S implies a victory if position P5 on the tracker is not a Spy victory.

Model Exploration

When creating the model for the game, it was unclear on how to proceed with adding the constraints since the core of the game did not only depend on if the player played a Resistance token or Spy token but also factors such as vote tracker and round won had to be kept in mind.

In the initial stages of creating the encoding, there were only constraints for the token themselves. Such as mapping if R or S is true or false. However, as mentioned above this proved to be futile when taking into account other variables like vote tracking (Z). Whilst it was still necessary to maintain a mapping of the Resistance and Spy token; ultimately a better approach was to create constraints of the game condition itself per round. That way the model could be assessed based upon which missions were won or lost per game depending on what tokens were played and also if the mission was rejected via the vote tracker.

For example:

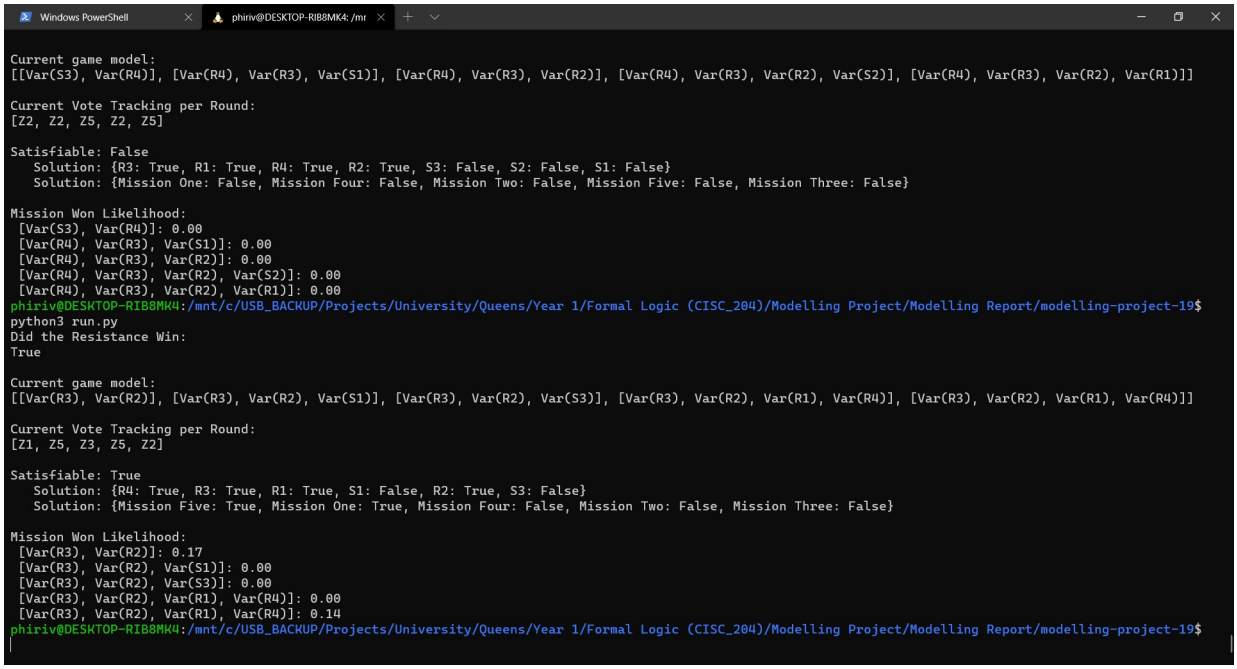
```
Current game model:
[[Var(S3), Var(S2)], [Var(R3), Var(R4), Var(R1)], [Var(R3), Var(R4), Var(R1)], [Var(R3), Var(R4), Var(R1), Var(R2)], [Var(R3),
Var(R4), Var(R1), Var(R2)]]

Current Vote Tracking per Round:
[Z4, Z5, Z4, Z1, Z5]

Satisfiable: True
   Solution: {R3: True, S2: False, R2: True, S3: False, R1: True, R4: True}
   Solution: {Mission Four: True, Mission One: False, Mission Two: False, Mission Three: True, Mission Five: False}
```

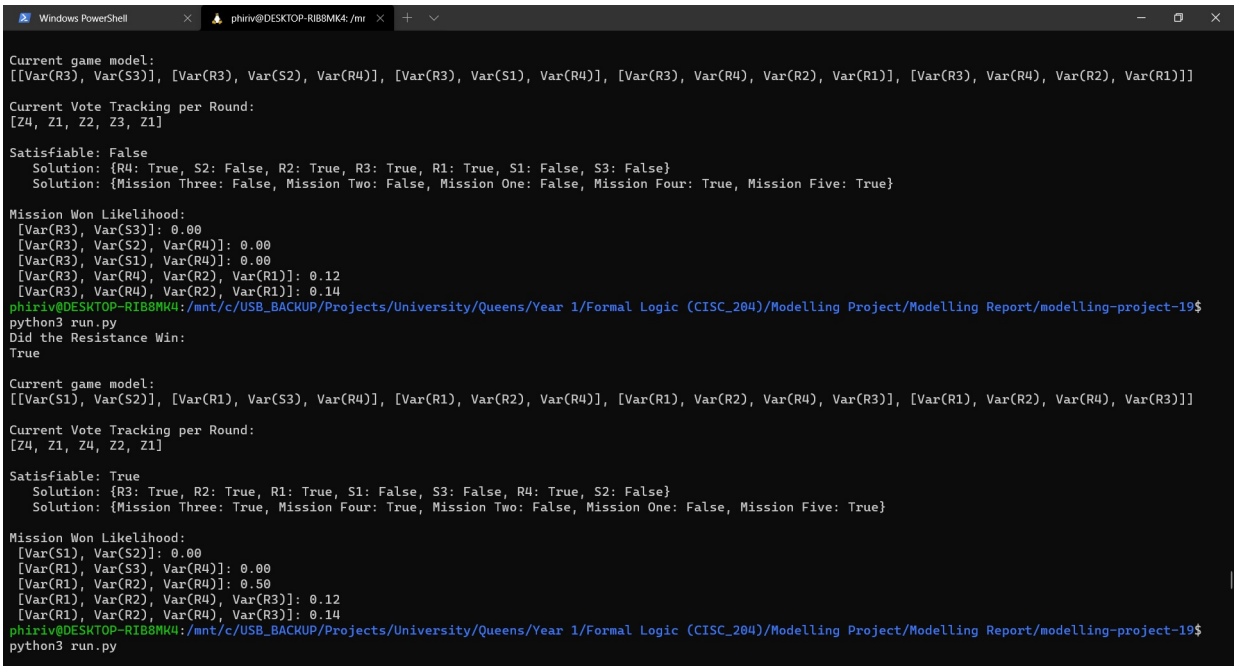
In this case for mission two, it is clear that since there exists R_1 & R_2 & R_3, mission two should map out to true. However, as observed from the vote tracking, the second round's mission was a failure as indicated by Z_5 in the vote tracker; therefore, mission two maps out to False rather than True.

Additionally, a small alteration was done to assess if the model is satisfiable. Because of the nature of the constraints the models would continuously assess to true despite having a clear hypothesis that only certain models should map out to true if the game is won which occurs when the resistance wins 3 rounds. Therefore with the creation of a function called ResistanceWin() (which assesses if the resistance wins a game) and using that information conjoined with the is_satisfiable() function was an alternative approach to examine if a model was satisfiable or not.

```
Did the Resistance Win:
False

Current game model:
[[Var(R1), Var(R2)], [Var(R1), Var(R2), Var(S1)], [Var(R1), Var(R2), Var(S2)], [Var(R1), Var(R2), Var(R4), Var(S3)], [Var(R1),
Var(R2), Var(R4), Var(R3)]]

Current Vote Tracking per Round:
[Z1, Z2, Z2, Z4, Z5]

Satisfiable: False
   Solution: {R3: True, R2: True, R4: True, R1: True, S1: False, S3: False, S2: False}
   Solution: {Mission Three: False, Mission Five: False, Mission Two: False, Mission Four: False, Mission One: True}
```

After the likelihood function was implemented using the dsharp solver included with the given library, repeated tests were done to ensure that the output made sense. Initially a problem occurred: the probabilities were exclusively 1.0 or 0.0 – either the Resistance was guaranteed a cakewalk or devastating loss! This binary (pun intended) outcome was mystifying, but was eventually discovered to be caused by using the wrong variable in the denominator of the fraction. After a quick fix the probabilities were proven to be in the proper range of between 0 to 1.0 via several tests (shown below). Interestingly, the odds of team R winning never went above 50% and were frequently below that, which indicates that according to our model the game is tilted in favour of the Spies. From this result it's evident that the game is structured to keep naively honest players on their toes, although more real-life testing is required to verify this.





Lastly, a final way the model was implemented was through a simple kind of automation. The various mission functions were coded so that (as per Prof. Muise's excellent advice), depending on what token was played the computer would make the most 'optimal' decision in choosing the team to go on the mission for the following rounds. This led to some interesting observations:

1. Since Mission Two and Mission Three have the same amount of tokens per their respective round, if the vote tracker for neither rounds is equal to Z_5, if Mission Two is true then in all cases Mission Three is true because the computer sends the same tokens from Mission Two onto Mission Three.

2. If Mission One and Mission Two are false, the vote tracker for neither round is equal to Z_5 and tokens S_1, S_2 and S_3 are mapped to false then it is true for all cases that Mission Three, Four and Five are true, and as a result the Resistance wins the entire game. This is because the model has already identified the 3 spies prior; hence, there is a guaranteed win for the next three rounds.

3. If Mission Five exists and Mission Four evaluates to true such that the vote tracker for round 4 is not equal to Z_5, it can be said that in all cases Mission Five will evaluate to true and the resistance will win the game. Hence, the model will be satisfiable and evaluates to true as well. This is because missions four and five contain the same number of player slots so the computer will automatically send all player tokens from Mission Four to Mission Five to guarantee a win for Mission Five. Additionally, the existence of Mission Five being true suggests that the game has not ended yet as none of the resistance or spy players have reached 3 points.

Therefore, a guaranteed win for Mission Five (or evaluation to True) suggests that the Resistance are guaranteed to win the game and the model will be satisfiable.

First Order Extension

Given the enumerative nature of our constraints in each voting round and mission, expanding the propositions to predicates is straightforward.

For a rejection to occur the conjunctions involving P_j are replaced with $\forall x : P(x)$, where a single vote against is represented by $\exists x : \neg P(x)$.

e.g. in round 1 ($i = 1$):
$\exists i : X(i) \wedge \forall x : P(x) \rightarrow \exists i : Y(i)$
$\exists i : X(i) \wedge \exists x : \neg P(x) \rightarrow \exists i : \neg Y(i)$

The 5 rejections in a row victory condition for the Spies directly follows:
$\forall y : Z(y) \rightarrow \forall k : \neg V(k)$

After mission approval, a single $\exists z : \neg Q(z)$ representing a played fail token causes the current mission to fail:
$\exists i : Y(i) \wedge \exists z : \neg Q(z) \rightarrow \exists i : \neg S(i)$

The final tally for the Resistance or Spies win condition is easily represented with existential quantifiers:
$\exists a : \exists b : \exists c : S(a, b, c) \rightarrow \forall k : V(k)$
$\exists d : \exists e : \exists f : \neg S(d, e, f) \rightarrow \forall k : \neg V(k)$

This radically simplifies things, as now it's no longer necessary to count every single different case where the Resistance wins 3 or more rounds to obtain an overall victory. Additional flexibility is gained for the possibility of increasing the number of rounds to an arbitrary number and subsequently changing the proportion of successes required for a given team to win the game.

□

Thank you for taking the time to read this lengthy report! All the best