OPTIMAL STRATEGIES FOR THE RESISTANCE

Liam Johansson
Princeton University
COS 521 Final Project
liamdj@princeton.edu

***K*eywords** hidden information · sequence form · counterfactual regret minimization

1 Background

1.1 Introduction

Games are important indicators of computer capabilities. Researchers have created bots capability of defeating any human in games such as in chess, go, and even complex, team-based games such as Starcraft 2 and DOTA 2. Hidden-information games, where certain data is only made known to a subset of the players, has received additional attention recently. In particular, poker has become a popular domain for research in solving large games of imperfect information. Another interesting example is The Resistance, which is a popular hidden roles game. This game has been featured in work such as developing a new form of Monte Carlo Tree Search, MT-ISMCTS [5], game AI development conferences [1], and applying deep neural networks to role deduction [11].

1.2 Extensive-form game notation

Games are often expressed in extensive-form, which is an intuitive way to consider a complex game. The following notion is similar to [6]. An extensive-form game consists of a tree, where each non-terminal *node* $h \in \mathcal{H}$ in the tree represents a distinct game history. Each node has is owned by a player or by 'chance'. Chance nodes have a fixed transition probability to each child. The *information sets* for player i partition the nodes owned by that player. An information set $I \in \mathcal{I}_i$ contains the set of game states that are indistinguishable to the acting player. *Terminal nodes* $z \in \mathcal{Z}$ have utility $u_i(z)$ for each player. In this paper, we restrict ourselves to *perfect recall* games, which means that players remember the sequence of actions leading to any node. In other words, there is a single path from every node to the root.

A *behavioral strategy* $\sigma_i : \mathcal{I} \to [0,1]^{|A(I)|}$ for a player gives a probability distribution over the actions $A(I)$ available in each information set I. Let $u(\sigma)$ be the expected utilities given than players use strategies σ. Denote $\pi^\sigma(h)$ to be the probability of reaching node h when players use strategies $\sigma = [\sigma_1, \sigma_2]$. Let $I[a]$ be the node reached from I after taking action a.

Given all other player's strategies, a *best response* is a strategy for the remaining player that maximizes their utility. A *Nash equilibrium* consists of strategies such that each is a best response to the other strategies, meaning that no player can improve their expected utility by modifying their own strategy. Any 2-player extensive-form games has an equilibrium. This is easy to see by expressing the game in *normal form*. Let all pure strategies for each player $A(I_1) \times \cdots \times A(I_{|\mathcal{I}|})$ be options in a bimatrix game, and

Table 1: Standard mission sizes. * indicates that the mission passes with only 1 failure.

Players	Mission number				
	1	2	3	4	5
5	2	3	2	3	3
6	2	3	4	3	4
7	2	3	3	4*	4
8	3	4	4	5*	5

let the entries of the payoff matrices A, B be $\sum_{z \in \mathcal{Z}} u_i(z)\pi^{\sigma}(z)$ where σ_1, σ_2 are the strategies given by the row and column indices, respectively. By Nash's theorem [9], the normal-form game has an equilibrium. Since all strategies in the extensive form game can be described by convex combinations of the pure strategies, the equilibrium in the corresponding bimatrix game induces an equilibrium in the extensive-form game.

2 The Resistance game

2.1 Rules

Similar to games like Mafia, Werewolf, or Secret Hitler, the players in The Resistance are randomly partitioned into two teams: the resistance and the spies. The spies know the identities of the other spies while the resistance members are unaware of the other players' teams, but have a majority. During each round, players successively propose missions, which consist of a subset of players. All players publicly vote on whether they approve the mission. A majority of players is required to approve the mission. When a mission is not approved, another player selects a new mission. When a mission is approved, each player on the mission secretly determines whether they will pass or fail the mission. Resistance members must always pass a mission. Missions pass when all members select pass, and fail otherwise. The spies win the game when three missions fail and the resistance wins when three missions pass, so the game lasts at most five rounds. The number of spies and members in each mission is predetermined. Table 1 shows those values for different player counts in a standard game [13].

2.2 Two-player extensive-form game

While each player in The Resistance acts individually and may have knowledge unknown to the other players, it is possible to formulate The Resistance as a 2-player zero-sum extensive-form game. Given some public game history (previous missions and their results), let there be some procedure to determine the next mission. Let us force resistance members to only propose the mission determined by that procedure and only approve that mission. This insures that in any given round, only that mission could be sent, since the resistance team has the majority. In this case, the spies must propose and vote exactly like they would if they were resistance, since otherwise they would reveal themselves for no strategic reason.

We may make the above simplifications since there is no private information that resistance members could benefit from sharing (unlike in Resistance: Avalon). While resistance members may have private information, for example when there is a two person mission that fails, there is nothing productive to be said to the other resistance members, since a spy would make the same public actions.

Now let us consider each team as a player. The resistance team must select a next mission based on the public history. The spy team knows which people are spies and must choose whether to fail a mission when that option is available. This defines an extensive-form game.

2.3 Equilibrium in the 5-person game

Theorem 2.1. *For 5-person Resistance, the following strategies are a Nash equilibrium with a 0.3 winrate for the resistance player:*

- *Resistance randomly selects 2 people to trust. Attempt 2-person missions with both people. Attempt 3-person missions with those plus each of the other people in turn.*
- *Spies pass 2-person missions and fail 3-person missions.*

Proof. For the resistance strategy, when randomly selecting 2 people, $\binom{3}{2}$ out of the $\binom{5}{2}$ possible spy locations have both people as resistance members. In those cases, both size 2 missions must pass and at least 1 of the size 3 missions must pass. This means that the resistance wins with at least 0.3 probability regardless of the spy strategy.

With the spy strategy, the size 2 missions always pass regardless of who is placed on them, so they can be ignored. The game is won for the resistance whenever they can correctly place the 3 resistance members on one of the 3 size 3 missions. When a mission fails, the resistance gains the information that that selection of 3 people has a spy, but learns nothing of the other size 3 subsets. The resistance can do no better than randomly guessing 3 unique size 3 subsets, which succeeds with probability $3/10$.

The above shows that neither side can improve their winrate against the other's strategy, so this is a Nash equilibrium. $\square$

2.4 Reducing the tree size

A first approach for constructing the game tree for P-person Resistance could be as follows:

- Chance randomly selects a subset of indices $[P]$ to be spies.
- Each resistance node has a child for each size m subsets of $[P]$, where m is the mission size for the current round.
- Each spy node has a pass and fail option if there is a spy on the mission.
- When three passing or failing missions are in the history, the node is terminal.

For 7-person resistance, such a game tree would have at least $\binom{4}{2}\binom{7}{2}\binom{7}{3}^3\binom{7}{4}^2 \approx 10^{10}$ nodes. This would probably be too big to feasibly solve on basic modern hardware in reasonable time. However, many of nodes in the above construction are redundant, so it is possible to produce an equivalent game with significantly fewer states. For the resistance player, the only distinguishing factor between players is which missions they have been on. Instead of considering all subsets for a mission, the resistance only needs to consider a single subsets for each strategically different option. Additionally, it is not necessary to reveal the exact indices of the spies to the spy player at the start of the game. The spy player only needs to know how many of the people sent on each selected mission are spies. Also, many actions can be eliminated from the tree, since they would never be chosen at equilibrium, such as re-picking a mission that has already failed or passing a mission when failing it would give the spies the win. These simplifications motivate the following game tree construction:

- Maintain a partition $\mathcal{P}$ of the indices $[P]$ such that people in the same subset have been on the exact same missions. Also maintain the number of spies in each of those subsets.
- Each resistance node has one child for each way of selecting m elements of $\mathcal{P}$ such that each subset can be repeated as many times as its size (in order words, one child for each strategically different mission). Remove options that contain a previously failed mission.
- After resistance picks a new mission, there may need to be a chance node that updates the positions of the spies. For each possible combination of spies compatible with the current

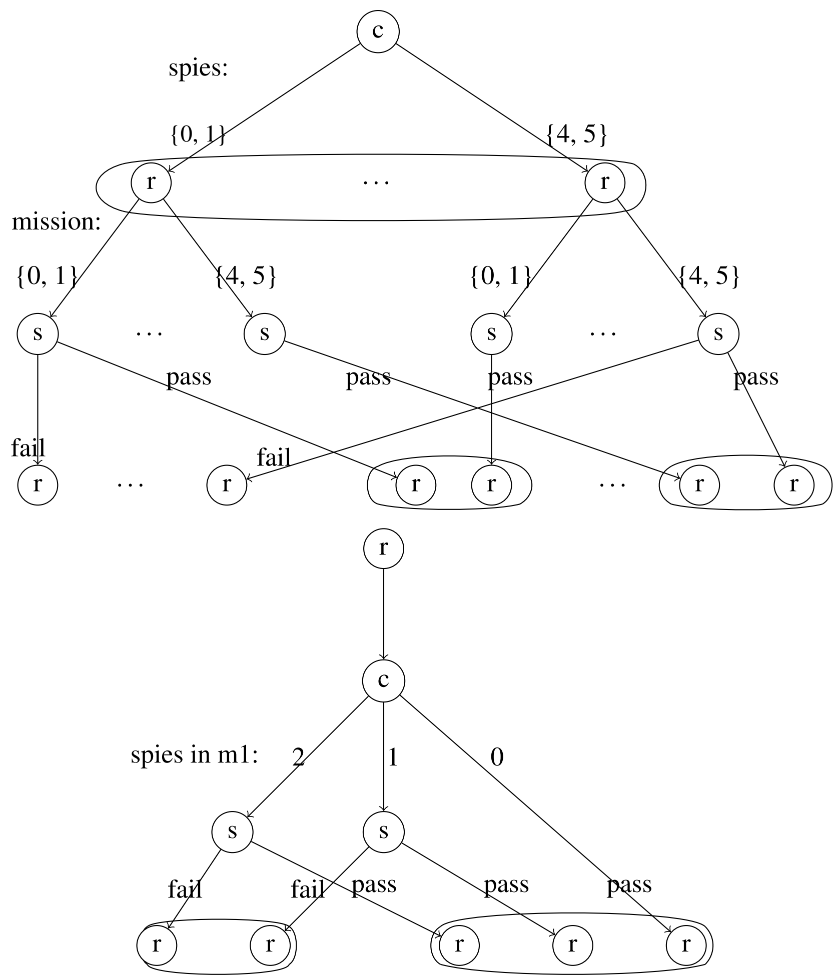

Figure 1: Start of initial tree versus reduced tree. Players are (r)esistance, (s)pies, and (c)hance. Circled nodes are in the same information set.

number of spies per subset, determine the spy placements in the new partition and transition with the appropriate probability.

- Spy nodes only exist when there is a spy on the mission and the number of passed missions is < 2 and the number of failed missions is < 2. Otherwise, the mission always passes when no spy is on the mission and fails when a spy is on the mission.

- When three passing or failing missions are in the history, the node is terminal.

The above construction leads to only $3.7 \cdot 10^5$ nodes for 7-person resistance, which should be feasible to solve on basic hardware.

3 Solving using sequence form

It is possible to construct a linear complementary problem whose solution gives an equilibrium of an extensive form game. This is the method described in [14].

3.1 Realization weights

We aim to express strategies in a way such that utilities are linear with the weight on each strategy. This is true of normal form strategies but not behavioral strategies. The issue with the normal form strategies is that there is exponentially many in the size of the game tree, since an action must be specified for each information set, even if the information cannot be reached due to the other actions selected. The solution is use sequences instead of pure strategies. A sequence $s_i \in \mathcal{S}_i$ consists of a possible list of actions taken by player i on some path in the game tree.

Selecting realization weights $r_i : \mathcal{S}_i \to [0, 1]$ is sufficient to express the strategy space for a player. The realization weight $r_i(s_i)$ represent the contribution of player i to $\pi_i(I)$, where I is an information set reachable using s_i. This is the probability of reaching I given that the other players always make moves compatible with reaching I. The correspondence with behavioral strategies is

$$r_i(s_i) = \prod_{a \in s_i} \sigma_i(a[I])(a)$$

where s_i contains the list of actions a taken by player i from information sets $a[I]$. Given the realization weights, the behavior strategies can be computed by

$$\sigma_i(s_i[I])(a) = \frac{r_i(s_i \cup \{a\})}{r_i(s_i)}$$

where $s_i[I]$ is the information set reached after s_i. When the realization weight of an information set is 0, the behavioral strategies are undefined, but this is irrelevant for computing a Nash equilibrium since those nodes can never be reached.

The realization weights must satisfy $r_i(\emptyset) = 1$ and $r_i(s_i) = \sum_{a \in s_i[I]} r_i(s_i \cup \{a\})$, so we have $1 + |\mathcal{I}_i|$ constraints. Let us index each possible sequence and let the realization weights be vectors $r_1 \in [0, 1]^{|\mathcal{S}_1|}$, $r_2 \in [0, 1]^{|\mathcal{S}_2|}$. Let E and F are matrices where rows correspond to information sets and columns correspond to sequences with entries in $\{-1, 0, 1\}$ that encode the above constraints for players 1 and 2, respectively. Then we can write that x, y are valid realization weights whenever

$$\begin{aligned} Ex &= [1 \quad 0 \quad \dots \quad 0]^\top = e \\ Fy &= [1 \quad 0 \quad \dots \quad 0]^\top = f \\ x, y &\geq 0 \end{aligned}$$

3.2 Equilibrium conditions

The utility of a player can be written as a linear function of their realization weights given their opponent's strategy. Define $A(s_1, s_2) = \sum_{z \in \mathcal{Z}} f^s(z) u_1(z)$ where $f^s(z)$ is the probability of ending at terminal node z when the players choose the actions along sequences s_1 and s_2. Note that if the sequences s_1 and s_2 are never terminal, then $A(s_1, s_2) = 0$. This allows us to write that

$$u_1(\sigma) = x^\top A y$$

where σ is the strategy given by realization weights x, y. This means that a best response to player 1's strategy x solves

$$\max_y (x^\top (-A)) y$$

$$\text{subject to } Fy = f, y \geq 0 \tag{1}$$

The dual LP is

$$\min_q q^\top f$$

$$\text{subject to } q^\top F \geq x^\top(-A) \tag{2}$$

Since a feasible solution is optimal only if the object values match by strong duality, we have that $x^\top(-A)y = q^\top f$, so

$$y^\top(A^\top x + F^\top q) = 0 \tag{3}$$

Similarly, the best response to player 2's strategy y solves

$$\max_x x^\top(Ay)$$

$$\text{subject to } Ex = e, x \geq 0 \tag{4}$$

which has a dual LP

$$\min_p p^\top f$$

$$\text{subject to } E^\top p \geq Ay \tag{5}$$

Applying strong duality gives that $x^\top Ay = e^\top p$, so

$$x^\top(-Ay + E^\top p) = 0 \tag{6}$$

x, y that solve equations 1 through 6 must be a Nash equilibrium, since the strategies are best responses to each other.

3.3 LCP formation

A *linear complementary problem* (LCP) consists of finding a z such that

$$\begin{gathered} z^\top(Mz + q) = 0 \\ Mz + q \geq 0 \\ z \geq 0 \end{gathered}$$

As shown in [7], computing an equilibrium for a sequence-form game as described above can be formulated as a LCP by letting

$$M = \begin{bmatrix} & -A & E^\top & -E^\top & & \\ -B^\top & & & & F^\top & -F^\top \\ -E & & & & & \\ E & & & & & \\ & -F & & & & \\ & F & & & & \end{bmatrix}, \quad q = \begin{bmatrix} 0 \\ 0 \\ e \\ -e \\ f \\ -f \end{bmatrix}$$

3.4 LCP solve attempts

Unfortunately, I was unsuccessful when attempting to solve the LCPs produced for The Resistance. As shown in [7], Lemke's method always works on the the LCP instances given by extensive-form games in theory. However, multiple implementations of Lemke's algorithms as well as other LCP solvers failed to produce accurate solutions. Specifically, the following methods were attempted on toy extensive-form game examples:

- Lemke's algorithm (from Siconos [8], OpenOut [10], and MATLAB [2])
- Other pivot methods (from Siconos)
- Iterative solvers (from Siconos)
- QP-reformation (from Siconos and CVXOPT [4])

The only method that produced a correct solution for the toy examples was the non-symmetric quadratic program (NSPQ) solver from Siconos. The NSPQ solver failed on larger inputs. This suggests that there is not bug in the way that the inputs were produced from the game, since it was possible to get a correct solution in some cases. This article [3] explains numerical issues with Lemke's algorithm, so it may be that the kind of LCPs produced by extensive form games lead to stability issues, but this would require further investigation.

4 CFR

Since I could not find a way to solve LCPs, another method of solving large extensive form games was required. Counterfactual regret minimization (CFR) is a method often used to approximate equilibrium for games too large for to be solved exactly.

4.1 Regret

CFR involves iteratively improving each player's strategies and was introduced in [15]. Let σ^t be the strategies after t iterations. Define the *average overall regret* to be

$$R_i^T = \frac{1}{T} \max_{\sigma_i^*} \sum_{t=1}^{T} (u_i(\sigma_i^*, \sigma_{-i}^t) - u_i(\sigma^t))$$

In other words, the average overall regret for player i is the average difference in utility between the best single strategy played for all iterations and the strategies actually played. Also define the *average strategy* to be

$$\bar{\sigma}_i^t(I)(a) = \frac{\sum_{t=1}^{T} \pi_i^{\sigma^t}(I)\sigma^t(I)(a)}{\sum_{t=1}^{T} \pi_i^{\sigma^t}(I)}$$

A key result about regret is that in a zero-sum 2-player game, if both player's average overall regret is $< \frac{\epsilon}{2}$ at iteration T, then each player can gain no more than ϵ utility by deviating from $\bar{\sigma}^T$, which means that the strategies are an ϵ-Nash equilibrium.

4.2 Counterfactual regret

By decomposing average overall regret into regret at each information set, it is possible to produce strategies converging to a Nash equilibrium using self-play. Let $\sigma|_{I \to a}$ be the strategy where action a is always playing at information set I. Define the *counterfactual regret* + to be

$$R_i^{T+}(I, a) = \frac{1}{T} \sum_{t=1}^{T} \pi_{-i}^{\sigma^t}(I) \max\{0, u_i(\sigma^t|_{I \to a}, I) - u_i(\sigma^t, I)\}$$

Table 2: Solving with CFR.

Player count	ϵ	winrate	iterations	iteration per second	tree size
5	0.002	0.030	11080	22.1	12162
6	0.002	0.332	11920	8.55	42438
7	0.002	0.257	1950	1.19	369869
8	0.003	0.160	500	0.213	4110657

Counterfactual regret corresponds to the amount of regret a player has from not playing a at I.This specific formula was first suggested by [12]. Let the strategies at each iteration be given by

$$\sigma_i^{T+1}(I)(a) = \frac{R_i^{T+}(I,a)}{\sum_{b \in A(I)} R_i^{T+}(I,b)}$$

In the case that the denominator is 0, let $\sigma_i^{T+1}(I)$ be uniform over actions. As shown in [15], this gives average strategies $\bar{\sigma}^t$ that converge to a ϵ-Nash equilibrium in $O(\frac{1}{\epsilon^2})$ iterations.

4.3 Results

Vanilla CFR was implemented and successfully found strategies close to a Nash equilibrium for 5, 6, and 7-person Resistance, as shown in table 2. Winrate refers to the fraction of the time the resistance player wins. ϵ refers to *exploitability*, which means to the maximum winrate decrease a player could face by when their opponents deviates from the average strategy. The exploitability is found by recursively calculating the best responses in subgames.

The strategies for spies the 5-person game match the solution 2.1, but the resistance strategy is different. The rule ‘fail exactly the missions with size >2’ seems to apply to the other player sizes as well.

5 Conclusions

Of the two main approaches taken for calculating optimal strategies, CFR was the successful. It appears that tools for solving extensive-form game LCPs are not readily available, or the author may be making a large error when attempting to use those tools. CFR was straightforward to implement, effective, and more intuitive than LCP solving.

5.1 Comparison with other results

As shown in 2.1, an optimal strategy for spies is to pass 2-person missions and fail 3-person missions (and propose missions and vote as if they were a resistance member). [5] produced bots that played 5-person Resistance and reported winrates that were much lower than 0.7 for the spies, which suggests that the spies act in a way that distinguishes them from resistance members. From personal experience playing the game, the resistance similarly wins much more often in person than the equilibrium winrates. People, especially those less experienced with hidden role games, find it difficult to interact with in the group in the same way when they are resistance versus spy.

5.2 Future work

Given access to these optimal solutions, it would be interesting to analyze the specific strategies to see if they shed any light on how to better play the game. Current, it is possible to view the action frequencies at a particular node, but there are no ways to visualize the strategies.

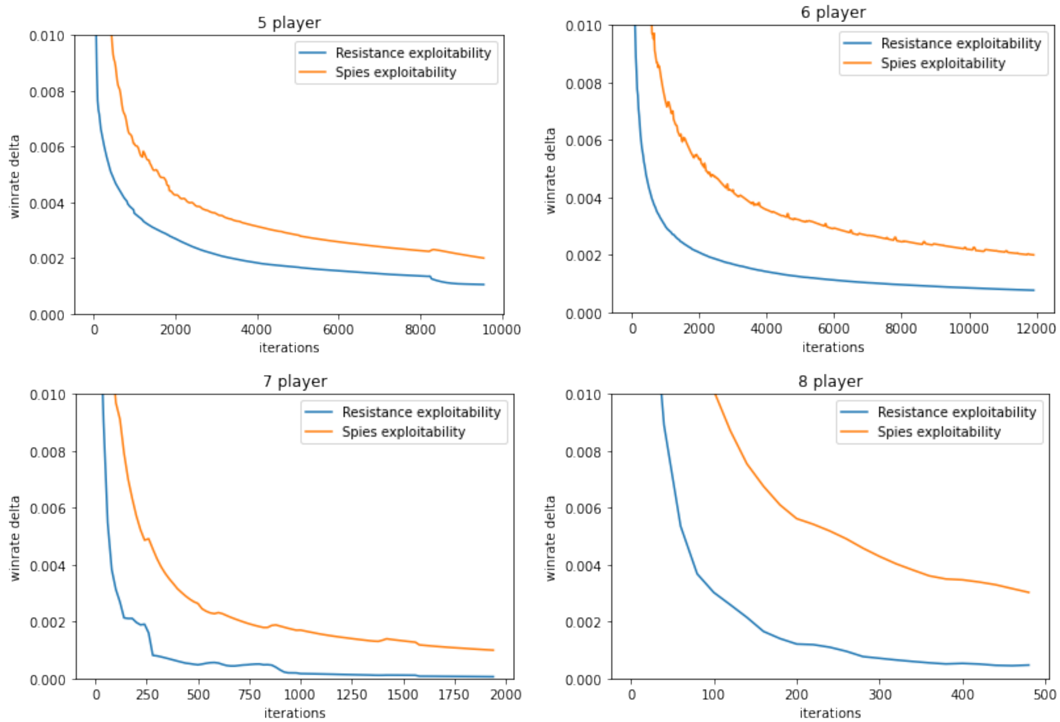

Figure 2: Exploitability versus number of iterations

Additionally, implementing a faster CFR algorithm, such as Monte Carlo CFR, may find the strategies faster, allowing less exploitable strategies to be found for the larger games. In addition, it may be possible to find exact equilibrium strategies by rounding. For instance in the 5-person game, the pure strategy 'fail when the mission size is 3 and pass when it is 2' is a spy strategy with exploitability 0. It seems possible that a similar result would emerge in the larger games.

References

[1] AiGameDev.com. *The Resistance AI*. URL: `https://github.com/aigamedev/resistance`.

[2] Andreas Almqvist. *A pivoting algorithm solving linear complementarity problems*. URL: `https://www.mathworks.com/matlabcentral/fileexchange/41485-a-pivoting-algorithm-solving-linear-complementarity-problems`.

[3] Oliver Brock, Jeff Trinkle, and Fabio Ramos. "A Numerically Robust LCP Solver for Simulating Articulated Rigid Bodies in Contact". In: (2009).

[4] *Cone Programming*. URL: `https://cvxopt.org/userguide/coneprog.html#quadratic-programming`.

[5] Peter I Cowling, Daniel Whitehouse, and Edward J Powley. "Emergent bluffing and inference with Monte Carlo tree search". In: *2015 IEEE Conference on Computational Intelligence and Games (CIG)*. IEEE. 2015, pp. 114–121.

[6] Michael Johanson et al. "Efficient Nash equilibrium approximation through Monte Carlo counterfactual regret minimization." In: *AAMAS*. Citeseer. 2012, pp. 837–846.

[7] Daphne Koller, Nimrod Megiddo, and Bernhard Von Stengel. "Efficient computation of equilibria for extensive two-person games". In: *Games and economic behavior* 14.2 (1996), pp. 247–259.

[8] *Linear Complementarity Problems (LCP)*. URL: `https://nonsmooth.gricad-pages.univ-grenoble-alpes.fr/siconos/users_guide/problems_and_solvers/lcp.html`.

[9] John Nash. "Non-cooperative games". In: *Annals of mathematics* (1951), pp. 286–295.

[10] *OpenOpt LCP Solver*. URL: `https://github.com/troyshu/openopt/blob/d15e81999ef09d3c9c8cb2fbb83388e9d3f97a98/openopt/solvers/HongKongOpt/LCPSolve.py`.

[11] Jack Serrino et al. "Finding friend and foe in multi-agent games". In: *arXiv preprint arXiv:1906.02330* (2019).

[12] Oskari Tammelin. "Solving large imperfect information games using CFR+". In: *arXiv preprint arXiv:1407.5042* (2014).

[13] *The Resistance (game)*. URL: `https://en.wikipedia.org/wiki/The_Resistance_(game)`.

[14] Bernhard Von Stengel. "Efficient computation of behavior strategies". In: *Games and Economic Behavior* 14.2 (1996), pp. 220–246.

[15] Martin Zinkevich et al. "Regret minimization in games with incomplete information". In: *Advances in neural information processing systems* 20 (2007), pp. 1729–1736.