Fairness Games

Let $G = ((S, E), (S_1, S_2))$ be a game graph, along with a set $P = \{(R_i, G_i)\}$ ($1 \leq i \leq d, R_i \subseteq S, G_i \subseteq S$). Each pair of sets $(R_i, G_i)$ represents a request and its corresponding grant.

Fairness and Co-Fairness Objectives

We define the following objectives:

- $\text{Streett}(P) = \{\pi \in \Pi | \forall i, 1 \leq i \leq d : \text{Inf}(\pi) \cap R_i \neq \emptyset \Rightarrow \text{Inf}(\pi) \cap G_i \neq \emptyset\}$
- $\text{Rabin}(P) = \{\pi \in \Pi | \exists i, 1 \leq i \leq d : \text{Inf}(\pi) \cap R_i \neq \emptyset \land \text{Inf}(\pi) \subseteq (S \setminus G_i)\}$

Streett objectives capture a fairness condition: whenever a request is made infinitely often, it must be granted infinitely often. For Rabin objectives, this condition is violated: some request is made infinitely often, but granted only finitely often.

Note that Streett objectives generalize parity objectives. Defining $P$ such that

$$G_1 \subset R_1 \subset G_2 \subset R_2 \ldots G_d \subset R_d$$

$p(s)$ for $s \in G_1 = 0$

$R_1 \setminus G_1 = 1$

$G_2 \setminus R_1 = 2$

$R_2 \setminus G_2 = 3$

$$\vdots$$

we get $\text{Streett}(P) = \text{Parity}(p)$.

Conversely, it is possible to reduce a fairness game on a graph $G$ with $n$ states, $m$ edges, and $d$ request-grant pairs to a parity game on a graph $G'$ with $n'$ nodes, $m'$ edges, and $d'$ priorities, where $n' = O(n \cdot m' \cdot d')$, $m' = O(m \cdot d' \cdot d^2)$, and $d' = O(d)$. (However, solving Streett games in this manner is not optimal.)

In general, winning strategies for Streett games require memory. Figure 1 depicts a simple fairness game in which player 1 clearly needs memory to win.
Memoryless Determinacy (of Rabin Games)

We begin by showing that, on assumption of determinacy, we can prove memoryless determinacy. In doing that, we will make use of the following property:

**Property 1.** Let \( \varphi \) be an objective. \( \varphi \) satisfies property 1 iff the following holds: for all plays \( \rho_1, \rho_2, \rho \), s.t. \( \text{Inf}(\rho) = \text{Inf}(\rho_1) \cup \text{Inf}(\rho_2) \), if \( \rho_1 \not\in \varphi \) \& \( \rho_2 \not\in \varphi \), then \( \rho \not\in \varphi \).

**Example 1: Streett Objectives.** Streett objectives do not satisfy this property.

Let \( P = \{(R_1, G_1), (R_2, G_2)\} \), and \( \text{Inf}(\rho_1) = \{R_1, G_2\}, \text{Inf}(\rho_2) = \{R_2, G_1\} \). We have \( \rho_1, \rho_2 \not\in \varphi \), but, since \( \text{Inf}(\rho) = \{R_1, R_2, G_1, G_2\}, \rho \in \varphi \).

**Example 2: Rabin Objectives.** Rabin objectives, on the other hand, do satisfy this property. Assume \( \rho_1, \rho_2, \rho \), s.t. \( \text{Inf}(\rho) = \text{Inf}(\rho_1) \cup \text{Inf}(\rho_2) \). Further,

\[
\rho_1 \not\in \text{Rabin}(P) \Rightarrow \forall i \text{ infinitely often } R_i \text{ implies infinitely often } G_i, \text{ and}
\rho_2 \not\in \text{Rabin}(P) \Rightarrow \forall i \text{ infinitely often } R_i \text{ implies infinitely often } G_i.
\]

Let \( R_i \cap \text{Inf}(\rho) \neq \emptyset \). Then \( R_i \) is visited infinitely often in either \( \rho_1 \) or \( \rho_2 \). As a consequence \( G_i \cap \text{Inf}(\rho_1) \neq \emptyset \) or \( G_i \cap \text{Inf}(\rho_2) \neq \emptyset \). Hence \( G_i \cap (\text{Inf}(\rho_1) \cup \text{Inf}(\rho_2)) = G_i \cap \text{Inf}(\rho) \neq \emptyset \), i.e. \( \rho \not\in \varphi \).

**Lemma.** For any objective \( \varphi \) that is determined and satisfies property 1, \( \varphi \) is memorylessly determined.

**Proof.** Without loss of generality, we will consider only game graphs where every node has at most two outgoing edges (one can always introduce auxiliary nodes to meet this requirement). We proceed by induction on the number \( k \) of player 1 nodes with two outgoing edges.

- For \( k = 0 \), player 1 has only one strategy, which is memoryless.
- Assume the lemma holds for game graphs with \( \leq k \) player 1 states with two outgoing edges. Take any game graph \( G \) with \( k + 1 \) such states, and let \( s \) be one of them. The situation is portrayed in figure 2. We have to consider two cases:
  - Player 1 wins in one of the subgames \( G_L \) or \( G_R \) that result from removing \( r \) or \( l \), respectively: wlog, assume she wins in \( G_L \). By the induction hypothesis, she has a memoryless winning strategy for \( G_L \), which can be extended to \( G \) by always choosing the left edge at \( s \).
  - Player 1 loses in both \( G_L \) and \( G_R \). Then player 2 has winning strategies \( \tau_L \) and \( \tau_R \) for \( G_L \) and \( G_R \), respectively. Take any player 1 strategy \( \sigma \) for \( G \). Consider the player 2 strategy \( \tau \) that behaves like \( \tau_L \) if player 1 last chose \( l \) at \( s \) and \( \tau_R \) if player 1 last chose \( r \). We claim that the play \( \pi(s, \sigma, \tau) \) is losing for player 1. To prove this, we need to look at three possible cases:
    (a) \( l \) is chosen only finitely often. In that case, after some time, we are in \( G_R \), and player 2 wins by playing \( \tau_R \) (via \( \tau \)).
    (b) \( r \) is picked only finitely often by player 1. That is, the game is eventually equivalent to \( G_L \), which is won by player 2 using \( \tau_L \) (via \( \tau \)).
    (c) both \( l \) and \( r \) are chosen infinitely often. The play \( \pi \) then looks as in figure 3 (assuming that player 1 decides to go left in the first move). \( \pi \) can be decomposed into plays \( \rho_L \) and \( \rho_R \) in \( G_L \) and \( G_R \), respectively. As a consequence, \( \text{Inf}(\pi) = \text{Inf}(\rho_L) \cup \text{Inf}(\rho_R) \). From \( \rho_1 \not\in \varphi \) and \( \rho_2 \not\in \varphi \), along with property 1, we can conclude that \( \pi \not\in \varphi \).
NP-Completeness of Rabin-Games

Given a game graph $G$ and a set of request-grant pairs $P$, the problem of deciding whether $s \in \langle\langle(1)\rangle\rangle Rabin(P)$ is NP-complete:

(a) If $s \in W_1$, a polynomial witness is given in the form of a memoryless strategy.

(b) Checking whether a given strategy is a witness can be done in polynomial time (see homework 3, problem 2).

(c) There is a polynomial reduction of 3-SAT to Rabin-Games: given a conjunction $\Phi$ of $m$ clauses over $n$ propositional variables $x_1, \ldots, x_n$, i.e. $\Phi = C_1 \land C_2 \land \ldots \land C_m$, where each literal $\xi_{ij} \in \{x_k, \bar{x}_k \mid 1 \leq k \leq n\}$ is either a variable or the negation of a variable, we construct a game graph $G(\Phi)$ as follows. For every literal and clause we introduce a player 1 state, along with a player 2 state $s_0$. $s_0$ has an outgoing edge to every node $c_i$ representing a clause. Every state $c_i$ has edges to the 3 literals $\xi_{ij}$ it contains. Finally, every literal has an outgoing edge to $s_0$. Figure 3 provides a sketch of the resulting game graph. Obviously, $G(\Phi)$’s size is polynomial in the size of $\Phi$.

As the objective we have Rabin($P$) for $P = \{(\{x_i\}, \{\bar{x}_i\}), (\{\xi_i\}, \{\bar{x}_i\}) \mid 1 \leq i \leq n\}$, i.e. paths $\pi$ satisfying

$$\bigvee_{i=1}^n [(x_i \in Inf(\pi) \land \bar{x}_i \notin Inf(\pi)) \lor (x_i \notin Inf(\pi) \land \bar{x}_i \in Inf(\pi))].$$

$\Phi$ is satisfiable if and only if player 1 has a winning strategy in $G(\Phi)$ (at $s_0$).

Proof. ($\Rightarrow$) Assume $\Phi$ is satisfiable, and let $V$ be a satisfying assignment. Then every clause $C_i$ must contain a literal $\xi_{ij}$ such that $V(\xi_{ij}) = 1$. By setting $\sigma(c_i) = \xi_{ij}$, one obtains a winning strategy for player 1: if $V(x_k) = 1$ for some $k$, then $\bar{x}_k$ is never visited, and vice versa.

($\Leftarrow$) Suppose player 1 has a winning strategy $\sigma$. We define a satisfying assignment $V$ for $\Phi$ as follows:

$$V(x_i) = \begin{cases} 1 & \text{if } \sigma(c_i) = x_j \text{ for some } i, \\ 0 & \text{otherwise}. \end{cases}$$

Note that it cannot be the case that $\sigma(c_j) = x_k$ and $\sigma(c_k) = \bar{x}_k$. Otherwise, player 2 could win against $\sigma$ by alternating visits between $c_j$ and $c_k$. Further, because $\sigma$ defines some outgoing edge for every (clause-)node, $V$ evaluates every clause $C_i$ to 1.

Determinacy of Streett/Rabin-Games

Finally, we will prove determinacy of Streett/Rabin-Games and provide an algorithm for computing the winning set for player 1.

Theorem. Let $G = ((S, E), (S_1, S_2))$ be a game graph and $P = \{(R_i, G_i) \mid 1 \leq i \leq d, R_i \subseteq S, G_i \subseteq S\}$. Then $\langle\langle(1)\rangle\rangle Streett(P) = S \setminus \langle\langle(2)\rangle\rangle Rabin(P)$.

Proof. We proceed by induction on $d$:
For $d = 1$, player 1 wins iff $\text{Inf}(R_1) \Rightarrow \text{Inf}(G_1)$. This is equivalent to a parity game on $G$ with parity function $p$ defined as

$$p(s) = \begin{cases} 
0 & \text{if } s \in G_1, \\
1 & \text{if } s \in R_1 \setminus G_1, \\
2 & \text{otherwise.}
\end{cases}$$

Determinacy follows from determinacy of parity games.

Suppose determinacy holds for games with $d < k$. Let $d = k$. We consider two cases:

1. For some $1 \leq i \leq k$, $G_i = \emptyset$. Wlog, suppose $G_k = \emptyset$. Consider the game graph $G'$ that results from removing the player 2 attractor $A = \langle\langle 2\rangle\rangle \Diamond R_k$ from $G$. Since $G_k = \emptyset$, $G'$ contains at most $k - 1$ request-grant pairs, and so by the induction hypothesis, $G'$ is determined. The situation is sketched on the left hand side of Figure 5. Let $W'_1$ be the winning set for player 1 in $G'$: since $G'$ only restricts player 1's strategies, $W'_1$ is winning for player 1 in $G$ as well. We remove its attractor $\langle\langle 1\rangle\rangle \Diamond W'_1$ from $G$, repeating these steps until $W'_1$ is empty. In the entire remaining game $G$, player 2 wins (Figure 5, right). We fix a strategy $\tau$ as follows:

$$\tau(s) = \begin{cases} 
\tau_A & \text{if } s \in \langle\langle 2\rangle\rangle \Diamond R_k, \\
\tau_W & \text{if } s \in S \setminus \langle\langle 2\rangle\rangle \Diamond R_k,
\end{cases}$$

where $\tau_A$ is the attractor strategy for $R_k$, and $\tau_W$ a winning strategy for $S \setminus \langle\langle 2\rangle\rangle \Diamond R_k$ (by determinacy of $G'$ and because $W'_1 = \emptyset$, this set is winning for player 2).

Let $s \in S$, and $\sigma$ an arbitrary strategy for player 1. $\pi(s, \sigma, \tau)$ is winning for player 2: if $A$ is visited infinitely often, $R_k$ is visited infinitely often, and because $G_k = \emptyset$, player 2 wins. Otherwise, if $A$ is visited only finitely often, after a finite number of moves, the game remains in $S \setminus A$, and again, player 2 wins (by playing $\tau_W$).

2. For $1 \leq i \leq k$, $G_i \neq \emptyset$. Let $A'_1 = \langle\langle 1\rangle\rangle \Diamond G_i$, and $G'$ be the subgame induced by $S \setminus A'_1$. Since $G_i = \emptyset$ in $G'$, it is determined by case (1). Let $W'_2$ be player 2’s winning set in $G'$. $\text{Attr}_2(W'_2) = \langle\langle 2\rangle\rangle \Diamond W'_2$ is winning for player 2 in the original game, too. We iteratively remove these player 2 winning regions from $G$ until $W'_2 = \emptyset$. The resulting game graph $G$ is depicted on the right hand side of Figure 6. In general, $G$ need not be winning for player 1. In particular, combining the strategies $\sigma'_W$ and $\sigma'_A$ does not necessarily
result in a winning strategy, simply because playing the attractor strategy may cause additional requests ($R_j$ in Figure 6).

Fortunately, we can address this problem by iteratively removing player 2 winning sets $W_i$ from $G$ for all $1 \leq i \leq k$. Figure 7 shows the corresponding pseudocode. In the remaining game graph $G'$, player 1 has a winning strategy $\sigma$: for $1 \leq i \leq d$, let $U_i$ denote the (sub)game as shown on the right hand side of Figure 6. $G'$ serves as the game board for all $U_i$. We define $\sigma_i$ as

$$
\sigma_i(s) = \begin{cases} 
\sigma_A^i & \text{if } s \in (\text{Attr}_1(G_i) \setminus G_i), \\
\sigma_W^i & \text{if } s \in (S \setminus \text{Attr}_1(G_i)). 
\end{cases}
$$

The strategy $\sigma(s)$ behaves like $\sigma_i(s)$, using a counter $i$ to keep track of what subgame is currently played. $i$ is initially set to 1, and incremented ($i \leftarrow i \mod d + 1$) every time $\sigma_i$ visits $G_i$. Take some $s \in S$, and an arbitrary player 2 strategy $\tau$. We show that the play $\pi(s, \sigma, \tau)$ is winning for player 1. There are two possibilities:

(a) From some point on, $\pi$ remains in the subgame $U_j$. That is, $\pi$ stays in $W_j$, where player 1 wins by playing $\sigma_W^j$.

(b) $\pi$ proceeds through all subgames $U_i$ infinitely often (the blue edges in Figure 8 are chosen infinitely often). In that case, every set $G_i$ is visited infinitely often, and player 1 wins.

Memory Requirement and Runtime

An upper bound for the memory required by $\sigma$ in case of $k$ request-grant pairs is given by the recursion

$$M(k) = k \cdot M(k-1) = k!$$

Player 1 needs memory for the counter ranging over $\{1, \ldots, k\}$, multiplied by the amount of memory required for each subgame.

For the runtime of the algorithm in Figure 7, one obtains

$$T(n, m, d) = n^2 \cdot d \cdot T(n, m, d-1) + O(m \cdot d)$$

$$= O(n^{2d} \cdot d! \cdot m).$$

In the worst case, the outer loop is executed $n$ times (because the algorithm removes at least one state in each iteration). The inner loop proceeds through all $d$ priorities, and for each one calls $n$ copies of itself (note the fixpoint computation in case (1)) to solve a subgame with $d-1$ priorities. The attractor computations in line 3 add up to $O(m \cdot d)$. (For comparison, the time required to solve fairness games by reduction to parity games is $(n \cdot d! \cdot d^2)^{O(d)}$.)
repeat
   for $i = 1$ to $d$ do
      $W^2_i \leftarrow$ winning set for player 2 in subgame $G \setminus \text{Attr}_1(G_i)$
      if $W^2_i \neq \emptyset$, remove $\text{Attr}_2(W^2_i)$ from $G$
   end for
until $W^2_i = \emptyset$ for all $1 \leq i \leq d$
return $S$

Figure 7: (Partial) Algorithm for $\langle\langle 1 \rangle\rangle\text{Streett}(P)$

Figure 8: Player 1’s winning strategy $\sigma$