

Complexity Bounds for Regular Games

Paul Hunter¹ and Anuj Dawar²

¹Institut für Informatik
Humboldt University, Berlin

²Computer Laboratory
University of Cambridge

Mathematical Foundations of Computer Science, 2005

- Regular games: History and Motivation
- Condition types: Translations and Succinctness
- Completeness results
- Consequences and further work

An **infinite game** (V, E, Win) consists of:

- Two players – Player 0 and Player 1
- An arena (V, E) , and
- A winning condition $\text{Win} \subseteq V^\omega$.

Player 0 and Player 1 alternately move a token around (V, E) for an infinite number of moves generating an infinite sequence of vertices $\pi \in V^\omega$.

Player 0 wins if and only if $\pi \in \text{Win}$.

An infinite game (V, E, Win) consists of:

- Two players – Player 0 and Player 1
- An arena (V, E) , and
- A winning condition $\text{Win} \subseteq V^\omega$.

Player 0 and Player 1 alternately move a token around (V, E) for an infinite number of moves generating an infinite sequence of vertices $\pi \in V^\omega$.

Player 0 wins if and only if $\pi \in \text{Win}$.

An infinite game (V, E, Win) consists of:

- Two players – Player 0 and Player 1
- An arena (V, E) , and
- A winning condition $\text{Win} \subseteq V^\omega$.

Player 0 and Player 1 alternately move a token around (V, E) for an infinite number of moves generating an infinite sequence of vertices $\pi \in V^\omega$.

Player 0 wins if and only if $\pi \in \text{Win}$.

A game (V, E, Win) is **regular** if there exists $\mathcal{F} \subseteq 2^V$ such that

$$\pi \in \text{Win} \iff \text{inf}(\pi) \in \mathcal{F}$$

where $\text{inf}(\pi)$ is the set of vertices occurring infinitely often in π .

Examples

- Muller games
- Parity games

A game (V, E, Win) is **regular** if there exists $\mathcal{F} \subseteq 2^V$ such that

$$\pi \in \text{Win} \iff \text{inf}(\pi) \in \mathcal{F}$$

where $\text{inf}(\pi)$ is the set of vertices occurring infinitely often in π .

Examples

- Muller games
- Parity games

Close connections with infinite automata

- Equivalent to infinite alternating automata
- Used to show equivalence of Muller, Rabin and Parity tree automata, giving
 - Complementation of languages defined by Rabin tree automata
 - Decidability of $S2S$, SnS , Muller acceptance, ...

Close connections with infinite automata

- Equivalent to infinite alternating automata
- Used to show equivalence of Muller, Rabin and Parity tree automata, giving
 - Complementation of languages defined by Rabin tree automata
 - Decidability of $S2S$, SnS , Muller acceptance, ...

Theorem

A regular game is:

- *Determined (Martin, 1975)*
- *Decidable (McNaughton, 1993)*

Decision Problem

Given a regular game (V, E, Win) and a starting position $v \in V$ does Player 0 win from v ?

Theorem

A regular game is:

- *Determined (Martin, 1975)*
- *Decidable (McNaughton, 1993)*

Decision Problem

Given a regular game (V, E, Win) and a starting position $v \in V$ does Player 0 win from v ?

Theorem

A regular game is:

- *Determined (Martin, 1975)*
- *Decidable (McNaughton, 1993)*

Decision Problem

Given a regular game (V, E, Win) and a starting position $v \in V$ does Player 0 win from v ?

Soufflé: Types of winning conditions

A **winning condition type** is a method to describe a winning condition.

$$\langle \mathcal{I}_{(V,E)}, \text{Acc}_{(V,E)} \rangle \text{ where } \text{Acc}_{(V,E)} \subseteq \mathcal{I}_{(V,E)} \times V^\omega$$

For regular games $\text{Acc}_{(V,E)} \subseteq \mathcal{I}_{(V,E)} \times 2^V$

Example

The Explicit winning condition type is one which explicitly lists a family of subsets of V

Instance: \mathcal{F} where $\mathcal{F} \subseteq 2^V$

Acceptance: $\inf(\pi) \in \mathcal{F}$

Soufflé: Types of winning conditions

A winning condition type is a method to describe a winning condition.

$$\langle \mathcal{I}_{(V,E)}, \text{Acc}_{(V,E)} \rangle \text{ where } \text{Acc}_{(V,E)} \subseteq \mathcal{I}_{(V,E)} \times V^\omega$$

For regular games $\text{Acc}_{(V,E)} \subseteq \mathcal{I}_{(V,E)} \times 2^V$

Example

The Explicit winning condition type is one which explicitly lists a family of subsets of V

Instance: \mathcal{F} where $\mathcal{F} \subseteq 2^V$

Acceptance: $\inf(\pi) \in \mathcal{F}$

Soufflé: Types of winning conditions

A winning condition type is a method to describe a winning condition.

$$\langle \mathcal{I}_{(V,E)}, \text{Acc}_{(V,E)} \rangle \text{ where } \text{Acc}_{(V,E)} \subseteq \mathcal{I}_{(V,E)} \times V^\omega$$

For regular games $\text{Acc}_{(V,E)} \subseteq \mathcal{I}_{(V,E)} \times 2^V$

Example

The Explicit winning condition type is one which explicitly lists a family of subsets of V

Instance: \mathcal{F} where $\mathcal{F} \subseteq 2^V$

Acceptance: $\inf(\pi) \in \mathcal{F}$

Soufflé: Types of winning conditions

A winning condition type is a method to describe a winning condition.

$$\langle \mathcal{I}_{(V,E)}, \text{Acc}_{(V,E)} \rangle \text{ where } \text{Acc}_{(V,E)} \subseteq \mathcal{I}_{(V,E)} \times V^\omega$$

For regular games $\text{Acc}_{(V,E)} \subseteq \mathcal{I}_{(V,E)} \times 2^V$

Example

The **Explicit** winning condition type is one which explicitly lists a family of subsets of V

Instance: \mathcal{F} where $\mathcal{F} \subseteq 2^V$

Acceptance: $\inf(\pi) \in \mathcal{F}$

Soufflé: Examples of winning condition types

Examples

Muller: Colours vertices and explicitly lists a set of subsets of the colours

Instance: (C, χ, \mathcal{F}) where $\chi : V \rightarrow C$ and $\mathcal{F} \subseteq 2^C$

Acceptance: $\chi(\text{inf}(\pi)) \in \mathcal{F}$

Parity: Assigns each vertex a priority and accepts sets with even minimal priority

Instance: χ where $\chi : V \rightarrow \omega$

Acceptance: $\min(\chi(\text{inf}(\pi)))$ is even

Win-set: Explicitly lists a family of subsets of $W \subseteq V$ and only considers the vertices in W

Instance: (W, \mathcal{F}) where $W \subseteq V$ and $\mathcal{F} \subseteq 2^W$

Acceptance: $\text{inf}(\pi) \cap W \in \mathcal{F}$

Emerson-Lei: Describes sets using a boolean formula with elements of V as atomic propositions

Soufflé: Examples of winning condition types

Examples

Muller: Colours vertices and explicitly lists a set of subsets of the colours

Instance: (C, χ, \mathcal{F}) where $\chi : V \rightarrow C$ and $\mathcal{F} \subseteq 2^C$

Acceptance: $\chi(\text{inf}(\pi)) \in \mathcal{F}$

Parity: Assigns each vertex a priority and accepts sets with even minimal priority

Instance: χ where $\chi : V \rightarrow \omega$

Acceptance: $\min(\chi(\text{inf}(\pi)))$ is even

Win-set: Explicitly lists a family of subsets of $W \subseteq V$ and only considers the vertices in W

Instance: (W, \mathcal{F}) where $W \subseteq V$ and $\mathcal{F} \subseteq 2^W$

Acceptance: $\text{inf}(\pi) \cap W \in \mathcal{F}$

Emerson-Lei: Describes sets using a boolean formula with elements of V as atomic propositions

Soufflé: Examples of winning condition types

Examples

Muller: Colours vertices and explicitly lists a set of subsets of the colours

Instance: (C, χ, \mathcal{F}) where $\chi : V \rightarrow C$ and $\mathcal{F} \subseteq 2^C$

Acceptance: $\chi(\text{inf}(\pi)) \in \mathcal{F}$

Parity: Assigns each vertex a priority and accepts sets with even minimal priority

Instance: χ where $\chi : V \rightarrow \omega$

Acceptance: $\min(\chi(\text{inf}(\pi)))$ is even

Win-set: Explicitly lists a family of subsets of $W \subseteq V$ and only considers the vertices in W

Instance: (W, \mathcal{F}) where $W \subseteq V$ and $\mathcal{F} \subseteq 2^W$

Acceptance: $\text{inf}(\pi) \cap W \in \mathcal{F}$

Emerson-Lei: Describes sets using a boolean formula with elements of V as atomic propositions

Soufflé: Examples of winning condition types

Examples

Muller: Colours vertices and explicitly lists a set of subsets of the colours

Instance: (C, χ, \mathcal{F}) where $\chi : V \rightarrow C$ and $\mathcal{F} \subseteq 2^C$

Acceptance: $\chi(\text{inf}(\pi)) \in \mathcal{F}$

Parity: Assigns each vertex a priority and accepts sets with even minimal priority

Instance: χ where $\chi : V \rightarrow \omega$

Acceptance: $\min(\chi(\text{inf}(\pi)))$ is even

Win-set: Explicitly lists a family of subsets of $W \subseteq V$ and only considers the vertices in W

Instance: (W, \mathcal{F}) where $W \subseteq V$ and $\mathcal{F} \subseteq 2^W$

Acceptance: $\text{inf}(\pi) \cap W \in \mathcal{F}$

Emerson-Lei: Describes sets using a boolean formula with elements of V as atomic propositions

Soufflé: Translations

Condition type \mathfrak{A} is **translatable** to condition type \mathfrak{B} if there is a polynomial time algorithm which, for every game of type \mathfrak{A} , produces a condition of type \mathfrak{B} that describes the same game.

Example

An explicitly presented game can be translated to a Muller game by taking the identity function as a colouring and using the same list of sets. Thus the Explicit condition type is translatable to the Muller condition type.

Example

Suppose we have a Muller game where half the vertices are coloured **blue**, half are **red**, and the list of sets is $\{\{\text{red}\}\}$. The explicit game equivalent to this requires exponentially more space to describe. Thus the Muller condition type is not translatable to the Explicit condition type.

Soufflé: Translations

Condition type \mathfrak{A} is translatable to condition type \mathfrak{B} if there is a polynomial time algorithm which, for every game of type \mathfrak{A} , produces a condition of type \mathfrak{B} that describes the same game.

Example

An explicitly presented game can be translated to a Muller game by taking the identity function as a colouring and using the same list of sets. Thus the Explicit condition type is translatable to the Muller condition type.

Example

Suppose we have a Muller game where half the vertices are coloured blue, half are red, and the list of sets is $\{\{red\}\}$. The explicit game equivalent to this requires exponentially more space to describe. Thus the Muller condition type is not translatable to the Explicit condition type.

Soufflé: Translations

Condition type \mathfrak{A} is translatable to condition type \mathfrak{B} if there is a polynomial time algorithm which, for every game of type \mathfrak{A} , produces a condition of type \mathfrak{B} that describes the same game.

Example

An explicitly presented game can be translated to a Muller game by taking the identity function as a colouring and using the same list of sets. Thus the Explicit condition type is translatable to the Muller condition type.

Example

Suppose we have a Muller game where half the vertices are coloured **blue**, half are **red**, and the list of sets is $\{\{\text{red}\}\}$. The explicit game equivalent to this requires exponentially more space to describe. Thus the Muller condition type is not translatable to the Explicit condition type.

If \mathfrak{A} is translatable to \mathfrak{B} but \mathfrak{B} is not translatable to \mathfrak{A} , we say \mathfrak{B} is **more succinct** than \mathfrak{A} .

Theorem

- *The Emerson-Lei type is more succinct than the Muller type*
- *The Muller type is more succinct than the Win-set type*
- *The Win-set type is more succinct than the Explicit type*

Corollary

- *Deciding Muller games reduces to deciding Emerson-Lei games*
- *Deciding Win-set games reduces to deciding Muller games*
- *Deciding Explicit games reduces to deciding Win-set games*

If \mathcal{A} is translatable to \mathcal{B} but \mathcal{B} is not translatable to \mathcal{A} , we say \mathcal{B} is more succinct than \mathcal{A} .

Theorem

- *The Emerson-Lei type is more succinct than the Muller type*
- *The Muller type is more succinct than the Win-set type*
- *The Win-set type is more succinct than the Explicit type*

Corollary

- *Deciding Muller games reduces to deciding Emerson-Lei games*
- *Deciding Win-set games reduces to deciding Muller games*
- *Deciding Explicit games reduces to deciding Win-set games*

If \mathcal{A} is translatable to \mathcal{B} but \mathcal{B} is not translatable to \mathcal{A} , we say \mathcal{B} is more succinct than \mathcal{A} .

Theorem

- *The Emerson-Lei type is more succinct than the Muller type*
- *The Muller type is more succinct than the Win-set type*
- *The Win-set type is more succinct than the Explicit type*

Corollary

- *Deciding Muller games reduces to deciding Emerson-Lei games*
- *Deciding Win-set games reduces to deciding Muller games*
- *Deciding Explicit games reduces to deciding Win-set games*

Theorem

Deciding Win-set games is PSPACE complete

Proof (Sketch).

- Membership in PSPACE follows from PSPACE algorithm for Emerson-Lei games (Nerode, Rummel, Yakhnis 1996)
- PSPACE hardness is shown with a reduction from QSAT (Quantified Satisfiability)

Theorem

Deciding Win-set games is PSPACE complete

Proof (Sketch).

- Membership in PSPACE follows from PSPACE algorithm for Emerson-Lei games (Nerode, Rummel, Yakhnis 1996)
- PSPACE hardness is shown with a reduction from QSAT (Quantified Satisfiability)

Theorem

Deciding Win-set games is PSPACE complete

Proof (Sketch).

- Membership in PSPACE follows from PSPACE algorithm for Emerson-Lei games (Nerode, Rummel, Yakhnis 1996)
- PSPACE hardness is shown with a reduction from QSAT (Quantified Satisfiability)

Main course: Reduction from QSAT

$$\Phi = \dots \forall x_1 \exists x_0. \phi$$



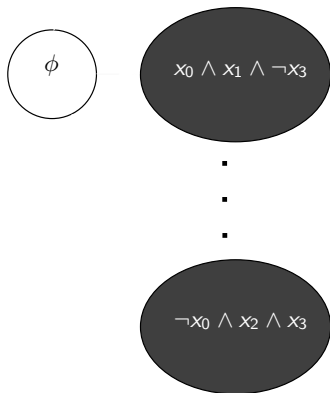
Main course: Reduction from QSAT

$$\Phi = \dots \forall x_1 \exists x_0. (x_0 \wedge x_1 \wedge \neg x_3) \vee \dots \vee (\neg x_0 \wedge x_2 \wedge x_3)$$



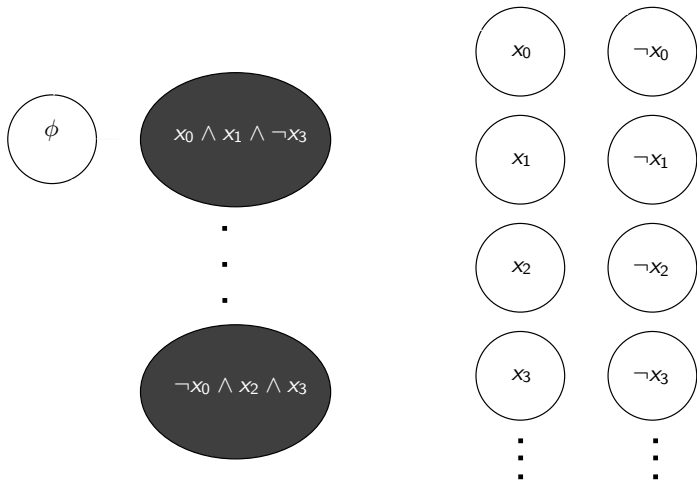
Main course: Reduction from QSAT

$$\Phi = \dots \forall x_1 \exists x_0. (x_0 \wedge x_1 \wedge \neg x_3) \vee \dots \vee (\neg x_0 \wedge x_2 \wedge x_3)$$



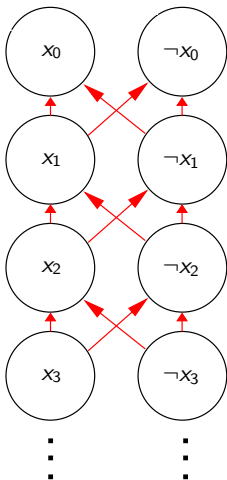
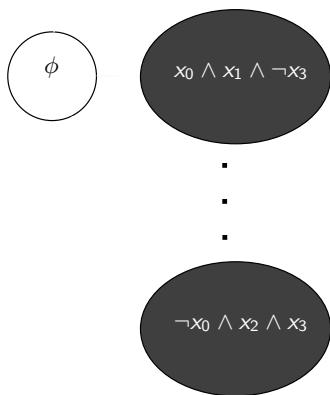
Main course: Reduction from QSAT

$$\Phi = \dots \forall x_1 \exists x_0. (x_0 \wedge x_1 \wedge \neg x_3) \vee \dots \vee (\neg x_0 \wedge x_2 \wedge x_3)$$



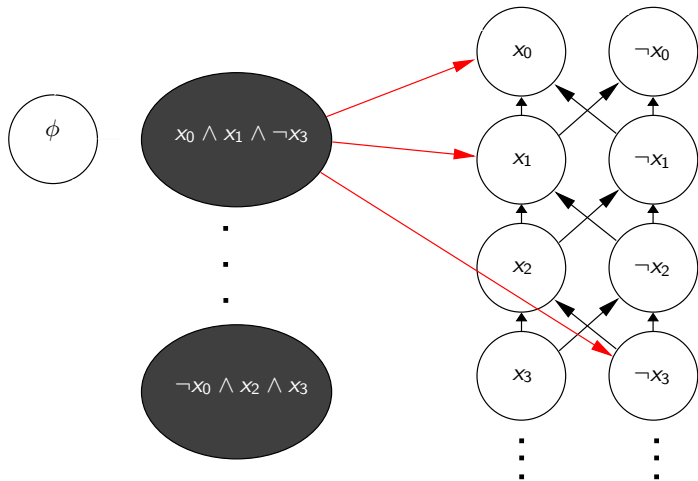
Main course: Reduction from QSAT

$$\Phi = \dots \forall x_1 \exists x_0. (x_0 \wedge x_1 \wedge \neg x_3) \vee \dots \vee (\neg x_0 \wedge x_2 \wedge x_3)$$



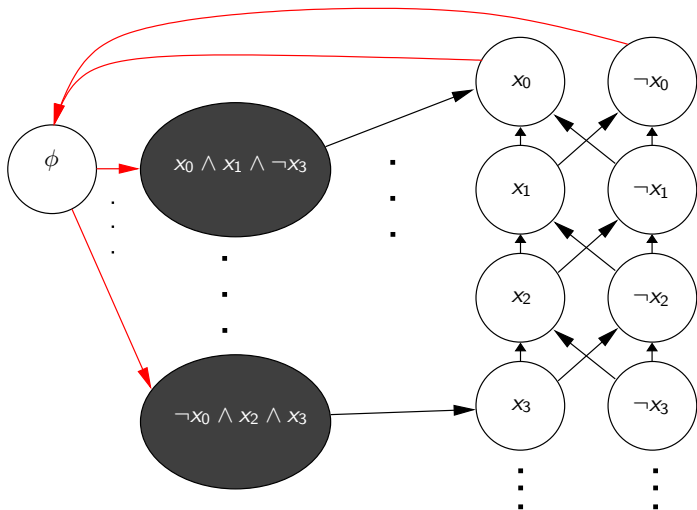
Main course: Reduction from QSAT

$$\Phi = \dots \forall x_1 \exists x_0. (x_0 \wedge x_1 \wedge \neg x_3) \vee \dots \vee (\neg x_0 \wedge x_2 \wedge x_3)$$



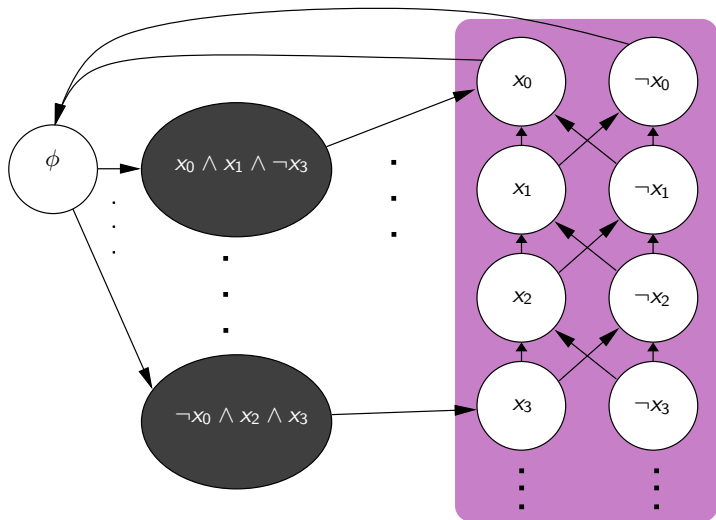
Main course: Reduction from QSAT

$$\Phi = \dots \forall x_1 \exists x_0. (x_0 \wedge x_1 \wedge \neg x_3) \vee \dots \vee (\neg x_0 \wedge x_2 \wedge x_3)$$



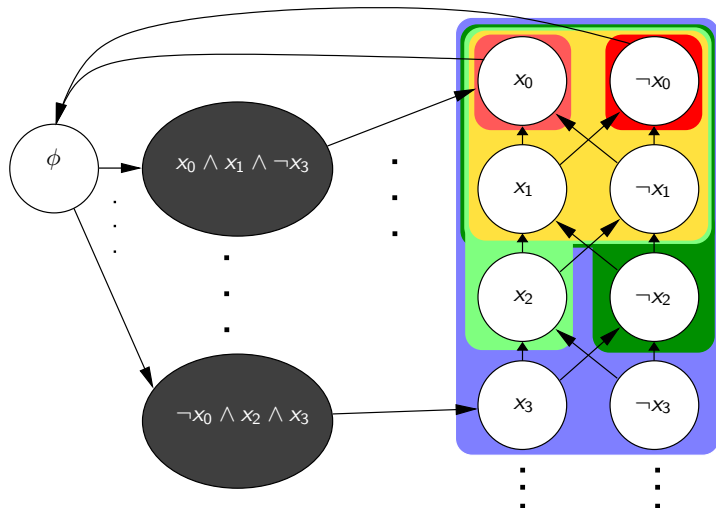
Main course: Reduction from QSAT

$$W = \{x_i, \neg x_i : 0 \leq i \leq n\}$$



Main course: Reduction from QSAT

$\mathcal{F} = \{S_i, S_i \cup \{x_i\}, S_i \cup \{\neg x_i\} : i \text{ even}\}$ where $S_i = \{x_j, \neg x_j : j < i\}$



Dessert: Further results

Corollary

Deciding Emerson-Lei games is PSPACE complete

Corollary

Deciding Muller games is PSPACE complete

Corollary

Deciding Muller games on arenas with bounded tree-width is PSPACE complete

Corollary

The non-emptiness and model-checking problems for Muller tree automata are PSPACE complete

Dessert: Further results

Corollary

Deciding Emerson-Lei games is PSPACE complete

Corollary

Deciding Muller games is PSPACE complete

Corollary

Deciding Muller games on arenas with bounded tree-width is PSPACE complete

Corollary

The non-emptiness and model-checking problems for Muller tree automata are PSPACE complete

Dessert: Further results

Corollary

Deciding Emerson-Lei games is PSPACE complete

Corollary

Deciding Muller games is PSPACE complete

Corollary

Deciding Muller games on arenas with bounded tree-width is PSPACE complete

Corollary

The non-emptiness and model-checking problems for Muller tree automata are PSPACE complete

We have:

- Examined winning conditions in isolation – introducing some new conditions and a notion of reduction
- Shown PSPACE completeness for Win-set games – a result which extends to decision problems associated with Muller tree automata.

Open problems:

- Complexity of explicitly presented games
- Complexity of Win-set games on arenas with bounded tree-width

We have:

- Examined winning conditions in isolation – introducing some new conditions and a notion of reduction
- Shown PSPACE completeness for Win-set games – a result which extends to decision problems associated with Muller tree automata.

Open problems:

- Complexity of explicitly presented games
- Complexity of Win-set games on arenas with bounded tree-width

We have:

- Examined winning conditions in isolation – introducing some new conditions and a notion of reduction
- Shown PSPACE completeness for Win-set games – a result which extends to decision problems associated with Muller tree automata.

Open problems:

- Complexity of explicitly presented games
- Complexity of Win-set games on arenas with bounded tree-width

We have:

- Examined winning conditions in isolation – introducing some new conditions and a notion of reduction
- Shown PSPACE completeness for Win-set games – a result which extends to decision problems associated with Muller tree automata.

Open problems:

- Complexity of explicitly presented games
- Complexity of Win-set games on arenas with bounded tree-width