

Games in logic

Games are used to capture "dynamics" of formulas. They offer, understanding of formula constructors in "operational" way. The three settings presented here focus on different constructors and in consequence use very different techniques.

- **Ehrenfeucht-Fraïssé games** focus on the behaviour of quantifiers. This leads to a notion of type and to the compositional method.
- **Parity games** are used to understand operators defined by fixpoints. This leads to theory of automata on infinite words or trees.
- **Game semantics** is used to understand dynamics of propositional constructs. This leads to accurate models of proofs and of programming languages.

Question: What are the connections between these three settings?

Ehrenfeucht-Fraïssé games and the composition method

Ehrenfeucht-Fraïssé games focus on the behaviour of quantifiers. This leads to a notion of type and to the compositional method. In consequence we study how to cut structures so that from the theory of parts one can obtain the theory of the whole. The other application is to obtain normal forms of formulas.

- Monadic second order logic.
- E-F games and the composition theorem for sums (Shelah's way).
- Computing MSO-theories of finite sequences and infinite sequences.
- CTL=Chain Logic.

[Thomas, "Ehrenfeucht Games, the Composition Method, and the Monadic Theory of Ordinal Words." Structures in Logic and Computer Science, LNCS 1261]

MSOL AND EHRENFUCHT-FRAÏSSÉ GAMES

MONADIC SECOND ORDER LOGIC

- Instead of quantification over elements we have quantification over sets.

$$\exists X.\varphi(X), \quad \forall X.\varphi(X)$$

- We have the inclusion predicate

$$X \subseteq Y$$

- Standard predicates can be "lifted" to sets:  
 $succ(X, Y), \quad X \leq Y, \quad X \subseteq P$

MSOL AND EHRENFUCHT-FRAÏSSÉ GAMES

MONADIC SECOND ORDER LOGIC

- Instead of quantification over elements we have quantification over sets.

$$\exists X.\varphi(X), \quad \forall X.\varphi(X)$$

- We have the inclusion predicate

$$X \subseteq Y$$

- Standard predicates can be "lifted" to sets:  
 $succ(X, Y), \quad X \leq Y, \quad X \subseteq P$

EHRENFUCHT-FRAÏSSÉ GAMES FOR MSOL

- We have two structures  $(A, \vec{P})$  and  $(B, \vec{Q})$  with some distinguished sets of elements.
- We also have  $\vec{k} = (k_1, \dots, k_n)$ , a vector of positive natural numbers.
- If  $\vec{k}$  is empty then **Duplicator** wins iff the two structures satisfy the same predicates with respect to  $\vec{P}$  and  $\vec{Q}$ . Otherwise **Spoiler** wins.
- If  $\vec{k} = \vec{\tau} \cdot m$  then Spoiler chooses one of the structures, say  $A$ , and  $m$  sets in this structure  $P_1, \dots, P_m$ . Duplicator replies by choosing  $Q_1, \dots, Q_m$  in  $B$  and then the  $\vec{\tau}$  game is played on  $(A, \vec{P}, P_1, \dots, P_m)$  and  $(B, \vec{Q}, Q_1, \dots, Q_m)$ .

E-F GAMES AND TYPES

QUANTIFIER ALTERNATION

- A formula of quantifier rank  $\vec{k} = (k_1, \dots, k_n)$  has the form:

$$\vec{Q}_n \vec{y}_n \dots \vec{Q}_1 \vec{y}_1. \varphi(\vec{x}, \vec{y}_n, \dots, \vec{y}_1)$$

where  $\vec{Q}_i \vec{y}_i$  stands for the vector of length  $k_i$  of quantifiers of the same type.

- Two structures  $(A, \vec{P})$  and  $(B, \vec{Q})$  are  $\vec{k}$ -equivalent, written  $(A, \vec{P}) \equiv^{\vec{k}} (B, \vec{Q})$  if they satisfy the same formulas of quantifier rank  $\vec{k}$ .
- **Fact:**  $(A, \vec{P}) \equiv^{\vec{k}} (B, \vec{Q})$  iff Duplicator has a winning strategy in the  $\vec{k}$  game on  $(A, \vec{P})$  and  $(B, \vec{Q})$ .

$\vec{k}$ -TYPES

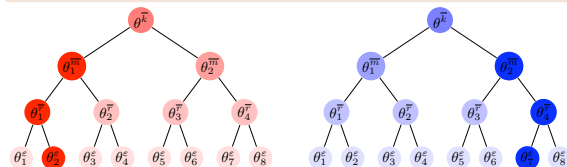
- $\text{Tp}^{\vec{k}}(A, \vec{P})$  is the set of qf formulas true in  $(A, P)$ .
- $\text{Tp}^{\vec{k}}(A, \vec{P})$ , for  $\vec{k} = \vec{\tau} \cdot m$



$$\{\text{Tp}^{\vec{\tau}}(A, \vec{P}, \vec{Q}) : \vec{Q} \subseteq |A|^m\}$$

IMPORTANT PROPERTY:  $(A, \vec{P}) \equiv^{\vec{k}} (B, \vec{Q})$  iff  $\text{Tp}^{\vec{k}}(A, \vec{P}) = \text{Tp}^{\vec{k}}(B, \vec{Q})$ .

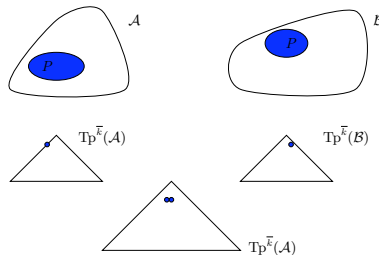
IMPORTANT PROPERTY:  $(A, \vec{P}) \equiv^{\vec{k}} (B, \vec{Q})$  iff  $\text{Tp}^{\vec{k}}(A, \vec{P}) = \text{Tp}^{\vec{k}}(B, \vec{Q})$ .



COMPOSITION THEOREMS

SUM

The  $\vec{k}$ -type of  $A + B$  is determined by (and can be computed from) the  $\vec{k}$  types of  $A$  and  $B$ .



COMPOSITION THEOREMS

SUM

The  $\vec{k}$ -type of  $A + B$  is determined by (and can be computed from) the  $\vec{k}$  types of  $A$  and  $B$ .

ORDERED SUM

The  $\vec{k}$ -type of  $\sum_{i \in \omega} A_i$  is determined by  $\vec{\tau}$ -type of the sequence  $(\text{type}^{\vec{k}}(A_0), \text{type}^{\vec{k}}(A_1), \dots)$ ; where  $\vec{k}$  and  $\vec{\tau}$  have the same length. (This sequence is over the alphabet  $\{Q_{\tau} : \tau \in \text{Types}^{\vec{k}}\}$ .)

## Deciding logics using compositional theorems

- A  $\bar{k}$ -type of a structure tells what  $\bar{k}$ -formulas are true in the structure.
- First, we will calculate possible types of finite sequences.
- Then, we calculate the types for infinite sequences.

## COMPUTING THE THEORY OF FINITE SEQUENCES

## THE THEORY OF FINITE SEQUENCES

- Consider structures  $\mathcal{A}_n = \langle \{1, \dots, n\}, \leq \rangle$ .
- Let  $\text{Tp}_{\text{fin}}^{\bar{k}} = \{\text{Tp}^{\bar{k}}(\mathcal{A}_n) : n \in \mathbb{N}\}$  (these are all possible  $\bar{k}$ -types of finite sequences)

COMPUTING  $\text{Tp}^{\bar{k}}(\text{Fin}(m))$ 

$$\begin{aligned} \text{Tp}^{\bar{k}}(\mathcal{A}_1) \\ \text{Tp}^{\bar{k}}(\mathcal{A}_2) &= \text{Tp}^{\bar{k}}(\mathcal{A}_1) + \text{Tp}^{\bar{k}}(\mathcal{A}_1) \\ &\vdots \\ \text{Tp}^{\bar{k}}(\mathcal{A}_{n+1}) &= \text{Tp}^{\bar{k}}(\mathcal{A}_n) + \text{Tp}^{\bar{k}}(\mathcal{A}_1) = \text{Tp}^{\bar{k}}(\mathcal{A}_j) \quad \text{for } j \leq n \end{aligned}$$

We take  $\text{Tp}_{\text{fin}}^{\bar{k}} = \bigcup_{i=1}^{\infty} \text{Tp}^{\bar{k}}(\mathcal{A}_i)$ .

## REMARK

$\text{Tp}_{\text{fin}}^{\bar{k}, m}$  gives us all possible  $\bar{k}$ -types of  $\{\langle \{1, \dots, n\}, \leq, P_1, \dots, P_m \rangle : \text{for } n \in \mathbb{N}\}$ .

$$\text{Tp}_{\text{fin}}^{\bar{k}, m} = \{\tau : \exists \sigma \in \text{Tp}_{\text{fin}}^{\bar{k}, m}, \tau \in \sigma\}$$
COMPUTING THE THEORY OF  $\langle \omega, \leq \rangle$ THE BASE STEP:  $\text{Tp}^m(\omega)$ 

We compute by hand the theory  $\text{Tp}^m(\omega)$  that is:

$$\{\text{Tp}^s(\omega, \bar{P}) : \bar{P} \in \mathcal{P}(\omega^m)\}$$

## OBSERVATION

If we can compute  $\text{Tp}^{\bar{k}}(\omega)$  then we can  $\text{Tp}^{\bar{k}}(\omega, \bar{Q})$  where each  $Q_i$  is either  $\emptyset$  or  $\omega$ .

INDUCTION STEP FOR  $\bar{k} \cdot m$ 

- $\text{Tp}^{\bar{k} \cdot m}(\omega) = \{\text{Tp}^{\bar{k}}(\omega, \bar{P}) : \bar{P} \in \mathcal{P}(\omega^m)\}$
- Each  $\text{Tp}^{\bar{k}}(\omega, \bar{P})$  can be presented as  $\tau + \sum_{\omega} \sigma$  for  $\tau, \sigma \in \text{Tp}_{\text{fin}}^{\bar{k}, m}$ .
- Computing  $\sum_{\omega} \sigma$  reduces to computing  $\text{Tp}^{\bar{k}}(\omega, \bar{Q})$  where only one  $Q_i = \omega$  and the rest is  $\emptyset$ .

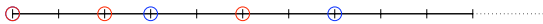
## RAMSEY ARGUMENT ON TYPES

## WE WANT TO COMPUTE:

$$\text{Tp}_{\text{fin}}^{\bar{k}, m}(\omega) = \{\text{Tp}^{\bar{k}}(\omega, \bar{P}) : \bar{P} \in \mathcal{P}(\omega^m)\}$$

We show that each  $\text{Tp}^{\bar{k}}(\omega, \bar{P})$  can be presented as  $\tau + \sum_{\omega} \sigma$  for  $\tau, \sigma \in \text{Tp}_{\text{fin}}^{\bar{k}, m}$ .

- Consider  $\mathcal{A} = (\omega, \leq, \bar{P})$



- Each interval  $(i, j)$  has its own  $\bar{k}$ -theory.
- We have  $F_{\mathcal{A}} : \mathbb{N}^2 \rightarrow \text{Tp}_{\text{fin}}^{\bar{k}, m}$ .
- By Ramsey Theorem there is an infinite  $S \subseteq \mathbb{N}$  and  $\sigma \in \text{Tp}_{\text{fin}}^{\bar{k}, m}$  such that for all  $i, j \in S$ ,  $F_{\mathcal{A}}(i, j) = \sigma$
- So  $\text{Tp}^{\bar{k}}(\mathcal{A})$  can be presented as  $\tau + \sum_{\omega} \sigma$ ; where  $\tau = F(1, i)$  and  $i = \min(S)$ .
- In consequence, it is enough to compute all possible:  $\tau + \sum_{\omega} \sigma$  for  $\tau, \sigma \in \text{Tp}_{\text{fin}}^{\bar{k}, m}$ .

COMPUTING THE THEORY OF  $\langle \omega, \leq \rangle$  (CONT.)INDUCTION STEP FOR  $\bar{k} \cdot m$ 

- $\text{Tp}^{\bar{k} \cdot m}(\omega) = \{\text{Tp}^{\bar{k}}(\omega, \bar{P}) : \bar{P} \in \mathcal{P}(\omega^m)\}$
- Each  $\text{Tp}^{\bar{k}}(\omega, \bar{P})$  can be presented as  $\tau + \sum_{\omega} \sigma$  for  $\tau, \sigma \in \text{Tp}_{\text{fin}}^{\bar{k}, m}$ .
- Computing  $\sum_{\omega} \sigma$  reduces to computing  $\text{Tp}^{\bar{k}}(\omega, \bar{Q})$  where only one  $Q_i = \omega$  and the rest is  $\emptyset$ .

## ORDERED SUM

The  $\bar{k}$ -type of  $\sum_{i \in \omega} \mathcal{A}_i$  is determined by  $\bar{\tau}$ -type of the sequence  $(\text{Tp}^{\bar{k}}(\mathcal{A}_0), \text{Tp}^{\bar{k}}(\mathcal{A}_1), \dots)$ ; where  $\bar{k}$  and  $\bar{\tau}$  have the same length. (This sequence is over the alphabet  $\{Q_{\tau} : \tau \in \text{Types}^{\bar{k}}\}$ .)

## OBSERVATION

If we can compute  $\text{Tp}^{\bar{k}}(\omega)$  then we can  $\text{Tp}^{\bar{k}}(\omega, \bar{Q})$  where each  $Q_i$  is either  $\emptyset$  or  $\omega$ .

## PART IC

## Understanding logics on trees

- Composition theorems for trees permit to talk about properties of trees in terms of its paths.
- FO=CTL\* over finite binary trees.
- Variants of this argument work for infinite trees, or unranked trees, etc.

## A COMPOSITION THEOREM FOR TREES

## FIRST-ORDER THEORY OF FINITE BINARY TREES

- We use first-order logic over predicates  $x \leq y$ ,  $left(x)$ ,  $right(x)$ .
- We will not need vectorial ranks. So we write  $\text{Tp}^k(\mathcal{A})$  for the  $k$ -type of  $\mathcal{A}$ , where  $k \in \mathbb{N}$ .

FROM TREES TO PATHS:  $\text{exp}^k(t, v)$ 

$$\begin{aligned} \text{exp}(t, w_4) &= \\ &(\lambda(w_1), r, \text{Tp}^k(v_1 \downarrow)) \\ &(\lambda(w_2), l, \text{Tp}^k(v_2 \downarrow)) \\ &(\lambda(w_3), r, \text{Tp}^k(v_3 \downarrow)) \\ &(\lambda(w_4), l, \text{Tp}^k(w_4 \downarrow)) \end{aligned}$$

## THEOREM

The  $k+1$ -type of a finite binary tree  $t$  is determined by the set

$$\{\text{Tp}^k(\text{exp}^k(t, v)) : v \in t\}$$

## FO=CTL\* ON FINITE BINARY TREES

## THEOREM (HAFER &amp; THOMAS)

The expressive powers of FO and CTL\* on finite binary trees are the same

## REMARK

On infinite trees it is not the case as in FOL we cannot single out an infinite path.

## PROOF: WE EXPRESS FO-TYPES IN CTL\*

- We want to express  $\text{Tp}^{k+1}(t)$ .
- By composition theorem it suffices to know  $\{\text{Tp}^k(\text{exp}^k(t, v)) : v \in t\}$ . Recall that  $\text{exp}^k(t, v)$  are the sequences over  $\Sigma \times \{l, r\} \times \text{Tp}^k$ .
- For every  $\text{Tp}^k(\text{exp}^k(t, v))$  there is a first order formula defining sequences with this property.
- By Kamp theorem over sequences FOL=LTL, so we have an equivalent LTL formula  $\alpha$  (with predicates from  $\Sigma \times \{l, r\} \times \text{Tp}^k$ )
- By induction every type in  $\text{Tp}^k$  is expressible by a CTL\* formula.
- We convert  $\alpha$  to a CTL\* formula  $\hat{\alpha}$  by replacing predicates from  $\Sigma \times \{l, r\} \times \text{Tp}^k(t)$  by an appropriate CTL\* formulas.
- We write CTL\* formula of the form  $\bigwedge_{i \in I} E \hat{\alpha}_i \wedge A(\bigvee_{i \in I} \hat{\alpha}_i)$ , expressing  $\text{Tp}^{k+1}(t)$ .

## PART II

## Parity games and model checking

Parity games are used to understand operators defined by fixpoints. Such operators talk not about a model but about an unfolding of the model. Their semantics is captured by infinite plays and parity condition.

- Introducing parity games via model-checking.
- Parity games  $\equiv$  model-checking of the mu-calculus.
- Where we can go from here: changing winning conditions, or changing the way games are played.

VERIFICATION (MODEL CHECKING)

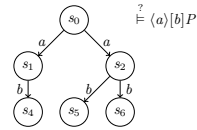
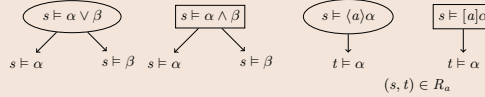
Given a transition system  $\mathcal{M}$  and a property  $\psi$ , check if  $\mathcal{M} \models \psi$

REFORMULATION

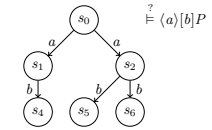
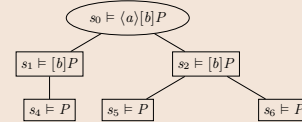
Construct a game  $G(\mathcal{M}, \psi)$  of two players: Adam and Eve.  
Fix the rules in such a way that

Eve wins from the initial position of  $G(\mathcal{M}, \psi)$  iff  $\mathcal{M} \models \psi$

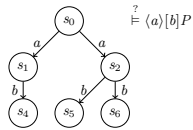
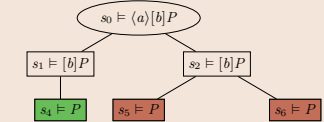
GAME RULES



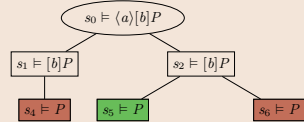
GAME



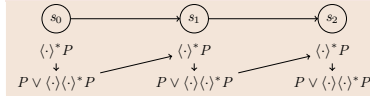
GAME



GAME



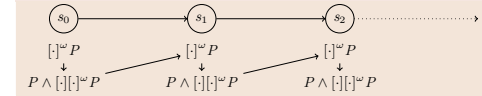
REACHABILITY:  $\langle \cdot \rangle^* P$



WHO WINS?

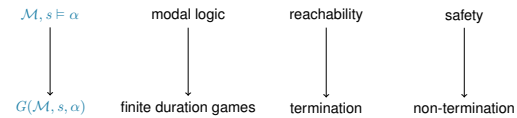
Eve wins if the game ends.

SAFETY:  $[\cdot]^\omega P$

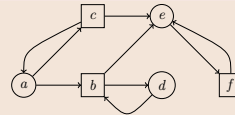


WHO WINS?

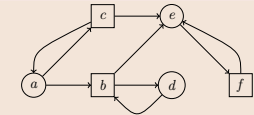
Eve wins if the game continues forever or ends because there is no successor.



DEFINITION (GAME  $\mathcal{G} = \langle V_E, V_A, R, \lambda : V \rightarrow C, Acc \subseteq C^\omega \rangle$ )



DEFINITION (GAME  $\mathcal{G} = \langle V_E, V_A, R, \lambda : V \rightarrow C, Acc \subseteq C^\omega \rangle$ )



DEFINITION (WINNING A PLAY)

Eve wins a play  $v_0 v_1 \dots$  iff the sequence is in  $Acc$ .

DEFINITION (WINNING POSITION)

A strategy for Eve is  $\sigma_E : V^* \times V_E \rightarrow V$ . A strategy is winning from a given position iff all the plays starting in this position and respecting the strategy are winning. A position is winning if there is a winning strategy from it.

**PROPERTIES**

- reachability
- safety
- etc.

**WINNING CONDITIONS**

- reachability:  $Acc = \{\text{sequences passing through a position from } F\}$ ,
- safety:  $Acc = \{\text{sequences of elements from } F\}$ ,
- repeated reachability:  $Acc = \{\text{sequences with infinitely many elements from } F\}$ ,
- ultimately safe:  $Acc = \{\text{almost all elements from } F\}$ .

THE MU-CALCULUS

**SYNTAX**

$$P \mid \neg P \mid X \mid \alpha \mid \alpha \vee \beta \mid \alpha \wedge \beta \mid (a)\alpha \mid [a]\alpha \mid \mu X.\alpha \mid \nu X.\alpha$$

**SEMANTICS**

Given  $\mathcal{M} = \langle V, \{E_a\}_{a \in Act}, P^M, \dots \rangle$  and  $Val : Var \rightarrow \mathcal{P}(V)$  we define  $[\alpha]_{Val}^M \subseteq \mathcal{P}(V)$ .

$$[P]_{Val}^M = P^M$$

$$[X]_{Val}^M = Val(X)$$

$$[(a)\alpha]_{Val}^M = \{v : \exists v'. E_a(v, v') \wedge v' \in [\alpha]_{Val}^M\}$$

$$[\mu X.\alpha(X)]_{Val}^M = \bigcap \{S \subseteq V : [\alpha(S)]_{Val}^M \subseteq S\}$$

Notation:  $\mathcal{M}, s \models \alpha$  for  $s \in [\alpha]_{Val}^M$ , where  $Val$  will be clear from the context.

We will give a characterization of the semantics in terms of games

EXAMPLE: REACHABILITY

REACHABILITY:  $\langle \cdot \rangle^* P \equiv \mu X.P \vee \langle \cdot \rangle X$

$\alpha \equiv \mu X.P \vee \langle \cdot \rangle X$

**DEFINITION (PARITY CONDITION:  $\Omega : V \rightarrow \{0, \dots, d\}$ )**

$$\vec{v} \in Acc \text{ iff } \liminf_{n \rightarrow \infty} \Omega(v_n) \text{ is even}$$

**OTHER CONDITIONS IN TERMS OF PARITY CONDITION**

- Infinitely often states from  $F \subseteq V$ .  
 $\Omega : V \rightarrow \{0, 1\}$  such that  $\Omega(v) = 0$  iff  $v \in F$ .
- Almost always states from  $F \subseteq V$ .  
 $\Omega : V \rightarrow \{1, 2\}$  such that  $\Omega(v) = 2$  iff  $v \in F$ .
- Reachability for  $F$ .  
Arrange so that each state from  $F$  is winning.
- Safety for  $F$ .  
 $\Omega(v) = 0$  for  $v \in F$  and arrange so that all states not in  $F$  are losing.

GAMES FOR THE MU-CALCULUS

**SETUP**

- We are given a transition system  $\mathcal{M}$  and a formula  $\alpha_0$ .
- We define a game  $G(\mathcal{M}, \alpha_0)$  where Eve wins from  $(s_0 \models \alpha_0)$  iff  $\mathcal{M}, s_0 \models \alpha_0$ .

**GAME RULES**

In  $(s \models P)$  Eve wins iff  $s \in P^M$     In  $(s \models \neg P)$  Eve wins iff  $s \notin P^M$

WHAT TO DO WITH  $\mu X.\alpha(X)$  AND  $\nu X.\alpha(X)$ ?

EXAMPLE: REACHABILITY

REACHABILITY:  $\langle \cdot \rangle^* P \equiv \mu X.P \vee \langle \cdot \rangle X$

$\alpha \equiv \mu X.P \vee \langle \cdot \rangle X$

$P \vee \langle \cdot \rangle \alpha$

Parity games  $\equiv \mu$ -calculus model checking

GAME RULES

**GAME RULES**

In  $(s \models P)$  Eve wins iff  $s \in P^M$     In  $(s \models \neg P)$  Eve wins iff  $s \in P^M$

$s \models \mu X.\alpha(X)$      $s \models \nu X.\alpha(X)$

These two rules may be the source of infinite plays.

EXAMPLE: REACHABILITY

REACHABILITY:  $\langle \cdot \rangle^* P \equiv \mu X.P \vee \langle \cdot \rangle X$

$\alpha \equiv \mu X.P \vee \langle \cdot \rangle X$

$P \vee \langle \cdot \rangle \alpha$

EXAMPLE: REACHABILITY

REACHABILITY:  $\langle \cdot \rangle^* P \equiv \mu X. P \vee \langle \cdot \rangle X$

$\alpha \equiv \mu X. P \vee \langle \cdot \rangle X$

Eve wins if the game ends in  $P$ .  $\mu X. \alpha(X) = \bigcup_{r \in \text{Ord}} \mu^r X. \alpha(X)$

EXAMPLE: REACHABILITY

REACHABILITY:  $\langle \cdot \rangle^* P \equiv \mu X. P \vee \langle \cdot \rangle X$

$\alpha \equiv \mu X. P \vee \langle \cdot \rangle X$

Eve wins if the game ends in  $P$ .  $\mu X. \alpha(X) = \bigcup_{r \in \text{Ord}} \mu^r X. \alpha(X)$

EXAMPLE: REACHABILITY

REACHABILITY:  $\langle \cdot \rangle^* P \equiv \mu X. P \vee \langle \cdot \rangle X$

$\alpha \equiv \mu X. P \vee \langle \cdot \rangle X$

Eve wins if the game ends in  $P$ .  $\mu X. \alpha(X) = \bigcup_{r \in \text{Ord}} \mu^r X. \alpha(X)$

EXAMPLE: REACHABILITY

REACHABILITY:  $\langle \cdot \rangle^* P \equiv \mu X. P \vee \langle \cdot \rangle X$

$\alpha \equiv \mu X. P \vee \langle \cdot \rangle X$

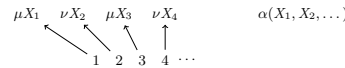
Eve wins if the game ends in  $P$ .  $\mu X. \alpha(X) = \bigcup_{r \in \text{Ord}} \mu^r X. \alpha(X)$

SAFETY:  $[\cdot]^\omega P \equiv \nu X. P \wedge [\cdot] X$

$\beta \equiv \nu X. P \wedge [\cdot] X$

Eve wins if the game continues for ever ends in  $P$ .

DEFINING WINNING CONDITIONS



- $\mu$ 's have odd ranks,
- $\nu$ 's have even ranks,
- if  $\beta$  is a subformula of  $\alpha$  then  $\beta$  has bigger rank than  $\alpha$ .

THE WINNING CONDITION IS THE PARITY CONDITION  
Eve wins if the smallest priority appearing infinitely often is even.

EXAMPLE  
 $\mu_1 Y. \nu_2 X. (P \wedge \langle \cdot \rangle X) \vee \langle \cdot \rangle Y$      $\nu_2 X. \mu_3 Y (P \wedge \langle \cdot \rangle X) \vee \langle \cdot \rangle Y$

MODEL CHECKING  $\equiv$  GAME SOLVING

MC  $\Rightarrow$  GAME SOLVING  
The problem  $\mathcal{M}, s_0 \models \alpha_0$  is reduced to deciding if Eve wins from the position  $(s_0 \models \alpha_0)$  in the game  $\mathcal{G}(\mathcal{M}, \alpha_0)$ .

GAME SOLVING  $\Rightarrow$  MC  
• Game can be represented as a transition system.  
• There is a  $\mu$ -calculus formula which is true exactly in positions where Eve wins.

REMARKS  
• Other program logics (with fixpoint definable operators) can be handled in the same way.  
• This approach permits also to handle satisfiability.  
• It also explains algorithmics of verification nicely. This is especially useful for verification of infinite structures.

PART IIC

Different ways of winning and playing games.

OTHER KINDS OF WINNING CONDITIONS

MEAN PAY-OFF GAME:  $G = \langle V_E, V_A, R, w : (V_E \cup V_A) \rightarrow \mathbb{N} \rangle$   
Outcome for Eve of a play  $v_0, v_1, \dots$  is:

$$\liminf_{n \rightarrow \infty} \frac{1}{n} \sum_{i=1}^n w(v_i).$$

For Adam it is  $\limsup$ .

DISCOUNTED PAYOFF GAME  $G = \langle V_E, V_A, R, w : (V_E \cup V_A) \rightarrow \mathbb{R} \rangle$

Outcome of  $v_0, v_1, \dots$  is  $(1 - \delta) \sum_{i=0}^{\infty} \delta^i w(v_i)$   
here  $0 < \delta < 1$  is a discount factor.

VALUE OF THE GAME  
Value of the game in a vertex  $v$  is a number  $\mathcal{V}_v$  such that:

- Eve has a strategy from  $v$  to have an outcome  $\geq \mathcal{V}_v$ , and
- Adam has a strategy from  $v$  to have an outcome  $\leq \mathcal{V}_v$ .

RESULTS ON PAYOFF GAMES

THEOREM (EHRENFEUCHT & MYCIELSKI)  
Every vertex of a mean payoff game has a value. Moreover the two players have positional optimal strategies.

THEOREM (ZWICK AND PATERSON)  
For every finite discounted payoff game the value exists in every vertex and is given as a unique solution of the set of equations:

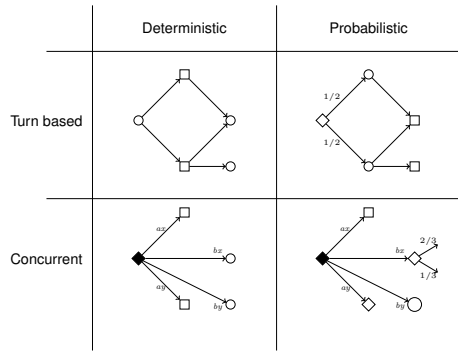
$$x_v = (1 - \delta)w(v) + \begin{cases} \max_{(v,u) \in R} \delta x_u & \text{if } v \in V_E \\ \min_{(v,u) \in R} \delta x_u & \text{if } v \in V_A \end{cases}$$

There are optimal positional strategies.

THEOREM (ZWICK & PATERSON)

When  $\delta \rightarrow 1$  then  $\mathcal{V}_\delta^{ZP}(v) \rightarrow \mathcal{V}^{EM}(v)$ .

MODIFYING RULES FOR PLAYING GAMES



PERFECT INFORMATION STOCHASTIC GAMES

**DEFINITION**  
Apart from positions for Eve and Adam there are randomized positions where a successor is chosen according to a probability distribution.

**EXAMPLE**

Adam wins in this game.

PERFECT INFORMATION STOCHASTIC GAMES, CONT.

**NONE OF THE PLAYERS MAY BE SURE TO WIN**

Eve wins with the probability 2/3 and Adam with the probability 1/3.

**THEOREM (DE ALFARO & MAJUMDAR, CHETTERJEE & JURDZINSKI & HENZINGER, ZIELONKA)**  
*In a finite game each state has a value and each player has an positional pure and optimal strategy.*

CONCURRENT GAMES

**DEFINITION**  
Two players choose their moves concurrently. Their joint choice determines the successor.

**EXAMPLE**

Eve's moves: delay,run,out  
Adam's moves: delay,throw,out

**OBSERVATION [DE ALFARO, HENZINGER]**  
There exists randomized strategies, but they may require infinite memory.

PART III

Game semantics

Game semantics is used to understand dynamics of propositional constructs. It extracts computational content from proofs, abstracting from irrelevant detail. This leads to accurate models of proofs and programming languages. As for E-F types, it is not the winning matters, but the ways to play.

- Proofs as games.
- Game semantics: an example.
- Brief summary of the results, and (potential) applications.

SATISFIABILITY IN PROPOSITIONAL LOGIC

**WE HAVE EXAMINED  $s \models \alpha$  NOW WE LOOK AT  $\alpha \vdash \beta$**

**GAME RULES**

**WHAT ONE CAN PROVE WITH THESE RULES?**  
Not much. These rules characterize  $\vee$  and  $\wedge$  in the free lattice.

SATISFIABILITY CONT.

**DIFFERENT WAYS OF PROVING  $\alpha_1 \wedge \alpha_2 \vdash \alpha_1 \vee \alpha_2$**

$\alpha_1 \wedge \alpha_2 \vdash \alpha_1 \vee \alpha_2$	$\alpha_1 \wedge \alpha_2 \vdash \alpha_1 \vee \alpha_2$
$\alpha_1 \wedge \alpha_2 \vdash \alpha_1$	$\alpha_1 \wedge \alpha_2 \vdash \alpha_2$
$\alpha_1 \vdash \alpha_1$	$\alpha_2 \vdash \alpha_2$

**THINGS WE CANNOT PROVE**  
 $(\alpha_1 \vee \alpha_2) \wedge \beta \vdash (\alpha_1 \wedge \beta) \vee (\alpha_2 \wedge \beta)$   
The normal solution is to go to sets of formulas on both sides.

SATISFIABILITY CONT.

**DIFFERENT WAYS OF PROVING  $\alpha_1 \wedge \alpha_2 \vdash \alpha_1 \vee \alpha_2$**

$\alpha_1 \wedge \alpha_2 \vdash \alpha_1 \vee \alpha_2$	$\alpha_1 \wedge \alpha_2 \vdash \alpha_1 \vee \alpha_2$
$\alpha_1 \wedge \alpha_2 \vdash \alpha_1$	$\alpha_1 \wedge \alpha_2 \vdash \alpha_2$
$\alpha_1 \vdash \alpha_1$	$\alpha_2 \vdash \alpha_2$

**STRATEGIES INSTEAD OF PROOFS**

$\alpha_1 \wedge \alpha_2 \vdash \alpha_1 \vee \alpha_2$	$\alpha_1 \wedge \alpha_2 \vdash \alpha_1 \vee \alpha_2$
$q_p$	$q_p$
$a_o$	$a_o$
$a_p$	$a_p$

PROOFS AS STRATEGIES

**STRATEGIES INSTEAD OF PROOFS**

$\vdash \alpha \rightarrow (\alpha \rightarrow \alpha)$	$\vdash \alpha \rightarrow (\alpha \rightarrow \alpha)$
$q_o$	$q_o$
$q_p$	$q_p$
$a_o$	$a_o$
$a_p$	$a_p$

**PROOFS AS PROGRAMS**

$\lambda x. \lambda y. y : \mathbb{N} \rightarrow \mathbb{N} \rightarrow \mathbb{N}$	$\lambda x. \lambda y. x : \mathbb{N} \rightarrow \mathbb{N} \rightarrow \mathbb{N}$
$q_o$	$q_o$
$q_p$	$q_p$
$a_o$	$a_o$
$a_p$	$a_p$

PROGRAM SEMANTICS AS A STRATEGY

$f : \mathbb{N} \rightarrow \mathbb{N} \vdash \text{if } (f(5) = 6) \text{ then } 7 \text{ else } 0 : \mathbb{N}$

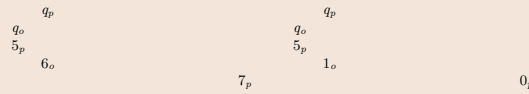


REMARKS

- Different strategies correspond to different programs.
- The semantics is not about provability.
- Semantics of a program is, roughly, a set of words: all plays permitted by the strategy.

PROGRAM SEMANTICS AS A STRATEGY

$f : \mathbb{N} \rightarrow \mathbb{N} \vdash \text{if } (f(5) = 6) \text{ then } 7 \text{ else } 0 : \mathbb{N}$       $f : \mathbb{N} \rightarrow \mathbb{N} \vdash \text{if } (f(5) = 6) \text{ then } 7 \text{ else } 0 : \mathbb{N}$



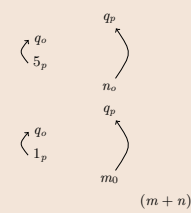
REMARKS

- Different strategies correspond to different programs.
- The semantics is not about provability.
- Semantics of a program is, roughly, a set of words: all plays permitted by the strategy.

MORE EXAMPLES

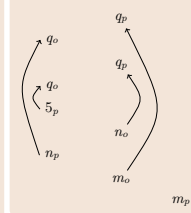
CALLING A FUNCTION TWICE

$f : \mathbb{N} \rightarrow \mathbb{N} \vdash f(5) + f(1) : \mathbb{N}$



NESTED CALLS

$f : \mathbb{N} \rightarrow \mathbb{N} \vdash \mathbb{N} \vdash f(f(5)) : \mathbb{N}$



IDEALIZED ALGOL

TYPES

$b ::= \text{com} \mid \text{exp} \mid \text{var} \quad \tau ::= b \mid \tau \rightarrow \tau$

RULES

$$\frac{}{\Gamma \vdash \text{skip}, \Omega : \text{com}} \quad \frac{i \in \{0, \dots, \text{max}\}}{\Gamma \vdash i : \text{exp}} \quad \frac{}{\Gamma, x : \tau \vdash x : \tau}$$

$$\frac{\Gamma, x : \tau \vdash M : \tau'}{\Gamma \vdash \lambda x. M : \tau \rightarrow \tau'} \quad \frac{\Gamma \vdash M : \tau \rightarrow \tau' \quad \Gamma \vdash N : \tau}{\Gamma \vdash MN : \tau'}$$

$$\frac{\Gamma \vdash M : \text{exp} \quad \Gamma \vdash M_0, M_1 : b}{\Gamma \vdash \text{ifzero } MM_0M_1 : b} \quad \frac{\Gamma \vdash M : \text{com} \quad \Gamma \vdash N : b}{\Gamma \vdash M; N : b}$$

$$\frac{\Gamma \vdash M : \text{var}}{\Gamma \vdash M : \text{exp}} \quad \frac{\Gamma \vdash M : \text{var} \quad \Gamma \vdash N : \text{exp}}{\Gamma \vdash M := N : \text{com}}$$

ITERATION AND RECURSION

$$\frac{\Gamma \vdash M : \text{exp} \quad \Gamma \vdash N : \text{com}}{\Gamma \vdash \text{while } M \text{ do } N : \text{com}} \quad \frac{\Gamma, x : \tau \vdash M : \tau}{\Gamma \vdash \mu x. M : \tau}$$

GAME SEMANTICS

OBSERVATIONAL EQUIVALENCE

Two terms are observationally equivalent,  $M \approx N$ , if they behave the same in all contexts

$$C[M] \Downarrow \text{skip} \quad \text{iff} \quad C[N] \Downarrow \text{skip}$$

- Quantification over all program contexts  $C[\cdot]$  makes this notion hard to work with.
- The notion is very compelling as it captures well semantical indistinguishability.

THEOREM (ABRAMSKY & MCCUSKER)

The game semantics of IA is fully abstract, which means that for all  $M$  and  $N$ :  $\llbracket M \rrbracket = \llbracket N \rrbracket$  iff  $M \approx N$ .

REMARKS

- For programs of type  $\mathbb{N} \rightarrow \mathbb{N}$  the semantics will be the input/output relation.
- The setting scales up to higher-order programs with free variables.
- The semantics is compositional.  
 $f : \mathbb{N} \rightarrow \mathbb{N} \vdash \text{if } (f(5) = 6) \text{ then } 7 \text{ else } 0 : \mathbb{N}$

REASONING ABOUT PROGRAMS USING GAME SEMANTICS

REASONING ABOUT PROGRAM EQUIVALENCE

- Construct the strategy representing a given program (typically using finite approximations for data domains).
- Sometimes this strategy can be represented by a finite automaton, or by a deterministic pushdown automaton.
- If so, we can check program equivalence by comparing languages for the two automata.

RESULTS

For Idealized Algol we have quite good understanding for which subclasses the equivalence problem is decidable.

CONCLUSIONS

- In E-F games, the semantics of a formula in a structure is represented by a type: a tree describing all potential extensions.
- In parity games, the semantics of a fixpoint formula is represented as an infinite unfolding of a formula. Winning conditions on infinite plays are needed to handle fixpoints of different types.
- In game semantics, formula, or program, is represented by a set of plays it admits. This set can be sometimes regular or context-free.

QUESTIONS

- Are there connections between these settings?
- Are there nontrivial results that can be transferred from one setting to another?