16: Hybrid Systems & Games
15-424: Foundations of Cyber-Physical Systems

André Platzer

aplatter@cs.cmu.edu
Computer Science Department
Carnegie Mellon University, Pittsburgh, PA
Outline

1. Learning Objectives
2. Motivation
3. Hybrid Games
   - Choices & Nondeterminism
   - Control & Dual Control
   - Hybrid Games
   - Differential Game Logic
   - Demon’s Controls
   - Operational Game Semantics
   - Filibusters & Finitude
4. Example: Robot Factory
5. Summary
Outline

1. Learning Objectives

2. Motivation

3. Hybrid Games
   - Choices & Nondeterminism
   - Control & Dual Control
   - Hybrid Games
   - Differential Game Logic
   - Demon’s Controls
   - Operational Game Semantics
   - Filibusters & Finitude

4. Example: Robot Factory

5. Summary
Learning Objectives
Hybrid Systems & Games

fundamental principles of computational thinking
logical extensions
PL modularity principles
compositional extensions
differential game logic
best-worst-case analysis
models of alternating computation

adversarial dynamics
conflicting actions
multi-agent systems
angelic/demonic choice

multi-agent state change
CPS semantics
reflections on choices
Outline

1. Learning Objectives

2. Motivation

3. Hybrid Games
   - Choices & Nondeterminism
   - Control & Dual Control
   - Hybrid Games
   - Differential Game Logic
   - Demon’s Controls
   - Operational Game Semantics
   - Filibusters & Finitude

4. Example: Robot Factory

5. Summary
CPS Analysis: Robot Control

Challenge (Hybrid Systems)

Fixed rule describing state evolution with both
- Discrete dynamics (control decisions)
- Continuous dynamics (differential equations)
CPS Analysis: Robot Control

Challenge (Hybrid Systems)

Fixed rule describing state evolution with both

- Discrete dynamics (control decisions)
- Continuous dynamics (differential equations)
CPS Analysis: Robot Control

Challenge (Games)

Game rules describing play evolution with both
- Angelic choices (player ♦ Angel)
- Demonic choices (player □ Demon)

<table>
<thead>
<tr>
<th></th>
<th>♦</th>
<th>Tr</th>
<th>Pl</th>
</tr>
</thead>
<tbody>
<tr>
<td>Trash</td>
<td>1,2</td>
<td>0,0</td>
<td></td>
</tr>
<tr>
<td>Plant</td>
<td>0,0</td>
<td>2,1</td>
<td></td>
</tr>
</tbody>
</table>
CPS Analysis: Robot Control

Challenge (Hybrid Games)

Game rules describing play evolution with

- Discrete dynamics (control decisions)
- Continuous dynamics (differential equations)
- Adversarial dynamics (Angel ♦ vs. Demon □)

André Platzer (CMU)
CPS Analysis: Robot Control

Challenge (Hybrid Games)

Game rules describing play evolution with
- Discrete dynamics (control decisions)
- Continuous dynamics (differential equations)
- Adversarial dynamics (Angel ♦ vs. Demon □)

\[ a_t \quad \omega_t \quad d_t \]

André Platzer (CMU)
CPS Analysis: RoboCup Soccer

Challenge (Hybrid Games)

Game rules describing play evolution with

- Discrete dynamics (control decisions)
- Continuous dynamics (differential equations)
- Adversarial dynamics (Angel 🏆 vs. Demon 🆚)

\[ a_1, a_2, a_3, a_4, a_5, a_6, a_7, a_8 \]

\[ d_{x_1}, d_{y_1}, d_{x_2}, d_{y_2}, d_{x_3}, d_{y_3}, d_{x_4}, d_{y_4} \]
CPSs are Multi-Dynamical Systems

CPS Dynamics
CPS are characterized by multiple facets of dynamical systems.

CPS Compositions
CPS combine multiple simple dynamical effects.

Tame Parts
Exploiting compositionality tames CPS complexity.
Dynamic Logics for Dynamical Systems

differential dynamic logic
\[ d\mathcal{L} = DL + HP \]

[α]φ  α  φ

differential game logic
\[ d\mathcal{GL} = GL + HG \]

⟨α⟩φ  φ

stochastic differential DL
\[ Sd\mathcal{L} = DL + SHP \]

quantified differential DL
\[ Qd\mathcal{L} = FOL + DL + QHP \]

JAR’08, CADE’11, LMCS’12, LICS’12

LICS’12, CADE’15, TOCL’15

André Platzer (CMU)
Outline

1 Learning Objectives

2 Motivation

3 Hybrid Games
   - Choices & Nondeterminism
   - Control & Dual Control
   - Hybrid Games
   - Differential Game Logic
   - Demon’s Controls
   - Operational Game Semantics
   - Filibusters & Finitude

4 Example: Robot Factory

5 Summary
Definition (Hybrid program $\alpha$)

\[
x := e \mid ?Q \mid x' = f(x) \& Q \mid \alpha \cup \beta \mid \alpha;\beta \mid \alpha^*
\]

Definition (dL Formula $P$)

\[
e \geq \bar{e} \mid \neg P \mid P \land Q \mid \forall x \ P \mid \exists x \ P \mid [\alpha]P \mid \langle \alpha \rangle P
\]
Differential Dynamic Logic $\mathcal{dL}$: Syntax

**Definition (Hybrid program $\alpha$)**

- $x := e | ?Q | x' = f(x) \& Q | \alpha \cup \beta | \alpha;\beta | \alpha^*$

**Definition ($\mathcal{dL}$ Formula $P$)**

- $e \geq \tilde{e} | \neg P | P \land Q | \forall x \ P | \exists x \ P | [\alpha] P | \langle \alpha \rangle P$
### Differential Dynamic Logic $\text{dL}$: Syntax

#### Definition (Hybrid program $\alpha$)

$$
\begin{align*}
  x &:= e \mid ?Q \mid x' = f(x) \& Q \mid \alpha \cup \beta \mid \alpha; \beta \mid \alpha^*
\end{align*}
$$

#### Definition ($\text{dL}$ Formula $P$)

$$
\begin{align*}
  e \geq \tilde{e} \mid \neg P \mid P \land Q \mid \forall x \ P \mid \exists x \ P \mid [\alpha]P \mid \langle \alpha \rangle P
\end{align*}
$$

---

**Nondeterminism during HP runs**

[Slide adapted from André Platzer (CMU)]
Differential Dynamic Logic $\mathcal{dL}$: Syntax

Definition (Hybrid program $\alpha$)

$x := e \mid ?Q \mid x' = f(x) \& Q \mid \alpha \cup \beta \mid \alpha;\beta \mid \alpha^*$

Definition ($\mathcal{dL}$ Formula $P$)

$e \geq \tilde{e} \mid \neg P \mid P \land Q \mid \forall x \ P \mid \exists x \ P \mid [\alpha]P \mid \langle \alpha \rangle P$

Nondeterminism during HP runs
### Differential Dynamic Logic $\mathcal{dL}$: Syntax

**Definition (Hybrid program $\alpha$)**

\[
x := e \mid ?Q \mid x' = f(x) \land Q \mid \alpha \cup \beta \mid \alpha ; \beta \mid \alpha^*
\]

**Definition ($\mathcal{dL}$ Formula $P$)**

\[
e \geq \bar{e} \mid \neg P \mid P \land Q \mid \forall x P \mid \exists x P \mid [\alpha]P \mid \langle \alpha \rangle P
\]

- **Differential Equation**
- **Nondet. Choice**
- **Nondet. Repeat**
- **Discrete Assign**
- **Test Condition**
- **Differential Equation**
- **Nondet. Choice**
- **Nondet. Repeat**
- **All Choices**
- **Some Choice**

André Platzer (CMU)
Differential Dynamic Logic $\mathcal{DL}$: Syntax

Definition (Hybrid program $\alpha$)

- Discrete Assign: $x := e$ | $?Q$
- Differential Equation: $x' = f(x) \& Q$
- Nondet. Choice: $\alpha \cup \beta$ | $\alpha; \beta$ | $\alpha^*$

Definition ($\mathcal{DL}$ Formula $P$)

- Nondet. Repeat: $e \geq \tilde{e}$ | $\neg P$ | $P \land Q$ | $\forall x \, P$ | $\exists x \, P$ | $[\alpha]P$ | $\langle \alpha \rangle P$
Game Operators

Angel Ops

- $\cup$  choice
- $\ast$  repeat
- $x' = f(x)$  evolve
- $?Q$  challenge

Let Angel be a player
Game Operators

- **Angel Ops**
  - $\cup$ choice
  - * repeat
  - $x' = f(x)$ evolve
  - $?Q$ challenge

- **Demon Ops**
  - $\cap$ choice
  - $\times$ repeat
  - $x' = f(x)^d$ evolve
  - $?Q^d$ challenge

Let Demon be another player.
Duality operator $d^d$ passes control between players
Game Operators

- **Angel Ops**
  - $\cup$ choice
  - $*$ repeat
  - $x' = f(x)$ evolve
  - $?Q$ challenge

- **Demon Ops**
  - $\cap$ choice
  - $\times$ repeat
  - $x' = f(x)^d$ evolve
  - $?Q^d$ challenge

Duality operator $^d$ passes control between players
Game Operators

◊ Angel Ops

\[ \cup \quad \text{choice} \]
\[ \ast \quad \text{repeat} \]
\[ x' = f(x) \quad \text{evolve} \]
\[ ?Q \quad \text{challenge} \]

◊ Demon Ops

\[ \cap \quad \text{choice} \]
\[ \times \quad \text{repeat} \]
\[ x' = f(x)^d \quad \text{evolve} \]
\[ ?Q^d \quad \text{challenge} \]

Duality operator \( d \) passes control between players
Definable Game Operators

**Diamond Angel Ops**

- \( \cup \)
- \( \ast \)
- \( x' = f(x) \)
- \(?Q\)
- choice
- repeat
- evolve
- challenge

**Diamond Demon Ops**

- \( \cap \)
- \( \times \)
- \( x' = f(x)^d \)
- \(?Q^d\)
- choice
- repeat
- evolve
- challenge

\[ d \]

\[ d \]
Differential Game Logic: Syntax

Definition (Hybrid game $\alpha$)

$$x := e \mid ?Q \mid x' = f(x) \mid \alpha \cup \beta \mid \alpha; \beta \mid \alpha^* \mid \alpha^d$$
### Definition (Hybrid game $\alpha$)

\[
x := e \mid ?Q \mid x' = f(x) \mid \alpha \cup \beta \mid \alpha; \beta \mid \alpha^* \mid \alpha^d
\]

### Definition (dGL Formula $P$)

\[
p(e_1, \ldots, e_n) \mid e \geq \tilde{e} \mid \neg P \mid P \land Q \mid \forall x \ P \mid \exists x \ P \mid \langle \alpha \rangle P \mid [\alpha] P
\]
Differential Game Logic: Syntax

Definition (Hybrid game $\alpha$)

- $x := e | ?Q | x' = f(x) | \alpha \cup \beta | \alpha ; \beta | \alpha^* | \alpha^d$

Definition (dGLC Formula $P$)

- $p(e_1, \ldots, e_n) | e \geq \bar{e} | \neg P | P \land Q | \forall x P | \exists x P | \langle \alpha \rangle P | [\alpha] P$
### Differential Game Logic: Syntax

**Definition (Hybrid game $\alpha$)**

- $x := e \mid ?Q \mid x' = f(x) \mid \alpha \cup \beta \mid \alpha;\beta \mid \alpha^* \mid \alpha^d$

**Definition (dGL Formula $P$)**

- $p(e_1, \ldots, e_n) \mid e \geq \bar{e} \mid \neg P \mid P \land Q \mid \forall x \, P \mid \exists x \, P \mid \langle \alpha \rangle P \mid [\alpha]P$

**Categories**
- Discrete Assign
- Test Game
- Differential Equation
- Choice Game
- Seq. Game
- Repeat Game
- Dual Game

**Terms**
- All Reals
- Some Reals

*TOCL’15*
Differential Game Logic: Syntax

Definition (Hybrid game $\alpha$)

\[ x := e \mid ?Q \mid x' = f(x) \mid \alpha \cup \beta \mid \alpha;\beta \mid \alpha^* \mid \alpha^d \]

Definition (dGL Formula $P$)

\[ p(e_1, \ldots, e_n) \mid e \geq \bar{e} \mid \neg P \mid P \land Q \mid \forall x \ P \mid \exists x \ P \mid \langle \alpha \rangle P \mid [\alpha]P \]

- Discrete Assign
- Test Game
- Differential Equation
- Choice Game
- Seq. Game
- Repeat Game
- Dual Game

- All Reals
- Some Reals
- Angel Wins
Differential Game Logic: Syntax

Definition (Hybrid game $\alpha$)

\[ x := e | ?Q | x' = f(x) | \alpha \cup \beta | \alpha; \beta | \alpha^* | \alpha^d \]

Definition (dGL Formula $P$)

\[ p(e_1, \ldots, e_n) | e \geq \bar{e} | \neg P | P \land Q | \forall x P | \exists x P | \langle \alpha \rangle P | [\alpha] P \]
Simple Examples

\[ \langle (x := x + 1; (x' = 1)^d \cup x := x - 1)^* \rangle (0 \leq x < 1) \]

\[ \langle (x := x + 1; (x' = x^2)^d \cup x := x - 1)^* \rangle (0 \leq x < 1) \]

\[ \langle (x := x + 1; (x' = 1)^d \cup (x := x - 1 \cap x := x - 2))^* \rangle (0 \leq x < 1) \]

\[ \langle (x := x + 1; (x' = x^2)^d \cup (x := x - 1 \cap x := x - 2))^* \rangle (0 \leq x < 1) \]
Simple Examples

\[ \langle x := x + 1; (x' = 1)^d \cup x := x - 1 \rangle^* \] (0 \leq x < 1)

\[ \langle x := x + 1; (x' = x^2)^d \cup x := x - 1 \rangle^* \] (0 \leq x < 1)

\[ \langle x := x + 1; (x' = 1)^d \cup (x := x - 1 \cap x := x - 2) \rangle^* \] (0 \leq x < 1)

\[ \langle x := x + 1; (x' = x^2)^d \cup (x := x - 1 \cap x := x - 2) \rangle^* \] (0 \leq x < 1)
Simple Examples

\[ \models \langle (x := x + 1; (x' = 1)^d \cup x := x - 1)^* \rangle (0 \leq x < 1) \]

\[ \models \langle (x := x + 1; (x' = x^2)^d \cup x := x - 1)^* \rangle (0 \leq x < 1) \]

\[ \langle (x := x + 1; (x' = 1)^d \cup (x := x - 1 \cap x := x - 2))^* \rangle (0 \leq x < 1) \]

\[ \langle (x := x + 1; (x' = x^2)^d \cup (x := x - 1 \cap x := x - 2))^* \rangle (0 \leq x < 1) \]
Simple Examples

\(\vdash \langle (x := x + 1; (x' = 1)^d \cup x := x - 1)^* \rangle (0 \leq x < 1)\)

\(\vdash \langle (x := x + 1; (x' = x^2)^d \cup x := x - 1)^* \rangle (0 \leq x < 1)\)

\(\not\vdash \langle (x := x + 1; (x' = 1)^d \cup (x := x - 1 \cap x := x - 2))^* \rangle (0 \leq x < 1)\)

\(\langle (x := x + 1; (x' = x^2)^d \cup (x := x - 1 \cap x := x - 2))^* \rangle (0 \leq x < 1)\)
Simple Examples

\[ \models \langle x := x + 1; (x' = 1)^d \cup x := x - 1 \rangle^* \rangle (0 \leq x < 1) \]

\[ \models \langle x := x + 1; (x' = x^2)^d \cup x := x - 1 \rangle^* \rangle (0 \leq x < 1) \]

\[ \not\models \langle x := x + 1; (x' = 1)^d \cup (x := x - 1 \cap x := x - 2) \rangle^* \rangle (0 \leq x < 1) \]

\[ \models \langle x := x + 1; (x' = x^2)^d \cup (x := x - 1 \cap x := x - 2) \rangle^* \rangle (0 \leq x < 1) \]
Simple Examples

≡ ⟨(x := x + 1; (x' = 1)^d ∪ x := x - 1)^*⟩ (0 ≤ x < 1)

≡ ⟨(x := x + 1; (x' = x^2)^d ∪ x := x - 1)^*⟩ (0 ≤ x < 1)

≰ ⟨(x := x + 1; (x' = 1)^d ∪ (x := x - 1 ∩ x := x - 2))^*⟩(0 ≤ x < 1)

≡ ⟨(x := x + 1; (x' = x^2)^d ∪ (x := x - 1 ∩ x := x - 2))^*⟩(0 ≤ x < 1)

since stuck at x = 0 which wins
Simple Examples

\[ \satisfies (x := x + 1; (x' = 1)^d \cup x := x - 1)^*(0 \leq x < 1) \]

\[ \satisfies (x := x + 1; (x' = x^2)^d \cup x := x - 1)^*(0 \leq x < 1) \]

\[ \not\satisfies (x := x + 1; (x' = 1)^d \cup (x := x - 1 \cap x := x - 2))^*(0 \leq x < 1) \]

\[ \satisfies (x := x + 1; (x' = x^2)^d \cup (x := x - 1 \cap x := x - 2))^*(0 \leq x < 1) \]

since stuck at \( x = 0 \) which wins
Simple Examples

\[ \models \langle (x := x + 1; (x' = 1)^d \cup x := x - 1)^* \rangle (0 \leq x < 1) \]

\[ \models \langle (x := x + 1; (x' = x^2)^d \cup x := x - 1)^* \rangle (0 \leq x < 1) \]

\[ \not\models \langle (x := x + 1; (x' = 1)^d \cup (x := x - 1 \cap x := x - 2))^* \rangle (0 \leq x < 1) \]

\[ \models \langle (x := x + 1; (x' = x^2)^d \cup (x := x - 1 \cap x := x - 2))^* \rangle (0 \leq x < 1) \]

since stuck at \( x = 0 \) which wins

\[ \not\models \langle (x := x + 1; (x' = x^2)^d \cup (x := x - 1 \cap x := x - 2))^* \rangle (0 < x \leq 1) \]
Simple Examples

\[ \models \langle (x := x + 1; (x' = 1)^d \cup x := x - 1)^* \rangle (0 \leq x < 1) \]

\[ \models \langle (x := x + 1; (x' = x^2)^d \cup x := x - 1)^* \rangle (0 \leq x < 1) \]

\[ \not\models \langle (x := x + 1; (x' = 1)^d \cup (x := x - 1 \cap x := x - 2))^* \rangle (0 \leq x < 1) \]

since stuck at \( x = 0 \) which wins

\[ \models \langle (x := x + 1; (x' = x^2)^d \cup (x := x - 1 \cap x := x - 2))^* \rangle (0 \leq x < 1) \]

since stuck at \( x = 0 \) which needs another iteration and then loses
Definable Game Operators

**Angel Ops**
- $\bigcup$ choice
- $\ast$ repeat
- $x' = f(x)$ evolve
- $?Q$ challenge

**Demon Ops**
- $\bigcap$ choice
- $\times$ repeat
- $x' = f(x)^d$ evolve
- $?Q^d$ challenge

$d$ transitions between Angel Ops and Demon Ops.
Definable Game Operators

- **Angel Ops**
  - $\bigcup$ choice
  - $\ast$ repeat
  - $x' = f(x)$ evolve
  - $?Q$ challenge

- **Demon Ops**
  - $\bigcap$ choice
  - $\times$ repeat
  - $x' = f(x)^d$ evolve
  - $?Q^d$ challenge

\[
\text{if}(Q)\alpha \text{ else } \beta \equiv \\
\text{while}(Q)\alpha \equiv \\
\alpha \cap \beta \equiv \\
\alpha \times \equiv \\
(x' = f(x) & Q)^d \quad x' = f(x) & Q \\
(x := e)^d \quad x := e \\
?q^d \quad ?Q
\]
Definable Game Operators

**Angel Ops**
- $\cup$
- $\ast$
- $x' = f(x)$
- $?Q$

**Demon Ops**
- $\cap$
- $\times$
- $x' = f(x)^d$
- $?Q^d$

- choice
- repeat
- evolve
- challenge

\[ \text{if}(Q) \alpha \text{ else } \beta \equiv (?Q; \alpha) \cup (?\neg Q; \beta) \]
\[ \text{while}(Q) \alpha \equiv \]
\[ \alpha \cap \beta \equiv \]
\[ \alpha \times \equiv \]
\[ (x' = f(x) \& Q)^d \quad x' = f(x) \& Q \]
\[ (x := e)^d \quad x := e \]
\[ ?Q^d \quad ?Q \]
Definable Game Operators

**Angel Ops**
- Union ($\cup$)
- Choice
- Repeat
- $x' = f(x)$
- Evolve
- ?$Q$
- Challenge

**Demon Ops**
- Intersection ($\cap$)
- Choice
- Repeat
- $x' = f(x)^d$
- Evolve
- $?Q^d$
- Challenge

**Formulas**
- \[ \text{if}(Q)\alpha\text{ else }\beta \equiv (?Q;\alpha) \cup (?\neg Q;\beta) \]
- \[ \text{while}(Q)\alpha \equiv (?Q;\alpha)^*;?\neg Q \]
- \[ \alpha \cap \beta \equiv \]
- \[ \alpha^\times \equiv \]
- \[ (x' = f(x) \land Q)^d \quad x' = f(x) \land Q \]
- \[ (x := e)^d \quad x := e \]
- \[ ?Q^d \quad ?Q \]
Definable Game Operators

**Angel Ops**

- $\cup$ choice
- $\ast$ repeat
- $x' = f(x)$ evolve
- $?Q$ challenge

**Demon Ops**

- $\cap$ choice
- $\times$ repeat
- $x' = f(x)^d$ evolve
- $?Q^d$ challenge

**Quantifier-Free Definable Operators**

- $\text{if}(Q)\alpha \text{ else } \beta \equiv (?Q; \alpha) \cup (?\neg Q; \beta)$
- $\text{while}(Q)\alpha \equiv (?Q; \alpha)^*; ?\neg Q$
- $\alpha \cap \beta \equiv$
- $\alpha \times \equiv$
- $(x' = f(x) \& Q)^d \quad x' = f(x) \& Q$
- $(x := e)^d \quad x := e$
- $?Q^d \quad ?Q$
Definable Game Operators

diamond Angel Ops

- $\cup$
- $\ast$
- $x' = f(x)$
- $\text{evolve}$
- $\text{challenge}$

diamond Demon Ops

- $\cap$
- $\times$
- $x' = f(x)^d$
- $\text{evolve}$
- $\text{challenge}$

if ($Q$) $\alpha$ else $\beta \equiv (\alpha; ?Q) \cup (\neg Q; \beta)$

while ($Q$) $\alpha \equiv (\alpha; ?Q)^\ast; ?\neg Q$

$\alpha \cap \beta \equiv (\alpha^d \cup \beta^d)^d$

$\alpha^\times \equiv$

$(x' = f(x) \& Q)^d$

$x' = f(x) \& Q$

$(x := e)^d$

$x := e$

$\text{challenge}$

$\text{challenge}$
Definable Game Operators

- **Angel Ops**
  - $\cup$
  - $\ast$
  - $x' = f(x)$
  - $?Q$
  - choice
  - repeat
  - evolve
  - challenge

- **Demon Ops**
  - $\cap$
  - $\times$
  - $x' = f(x)^d$
  - $?Q^d$
  - choice
  - repeat
  - evolve
  - challenge

- \(\text{if}(Q)\alpha \text{ else } \beta \equiv (?Q; \alpha) \cup (?\neg Q; \beta)\)
- \(\text{while}(Q)\alpha \equiv (?Q; \alpha)^* ; ?\neg Q\)
- \(\alpha \cap \beta \equiv (\alpha^d \cup \beta^d)^d\)
- \(\alpha^\times \equiv ((\alpha^d)^*)^d\)
- \((x' = f(x) \& Q)^d\)
- \((x := e)^d\)
- \(?Q^d\)

\(\equiv\)
Definable Game Operators

**Angel Ops**
- $\cup$
- $\ast$
- $x' = f(x)$
- $?Q$
- choice
- repeat
- evolve
- challenge

**Demon Ops**
- $\cap$
- $\times$
- $x' = f(x)^d$
- $?Q^d$
- choice
- repeat
- evolve
- challenge

\[
\begin{align*}
\text{if}(Q)\alpha \text{ else } \beta & \equiv (?Q; \alpha) \cup (?\neg Q; \beta) \\
\text{while}(Q)\alpha & \equiv (?Q; \alpha)^\ast; ?\neg Q \\
\alpha \cap \beta & \equiv (\alpha^d \cup \beta^d)^d \\
\alpha^\times & \equiv (((\alpha^d)^\ast)^d \\
(x' = f(x) & Q)^d & \neq x' = f(x) & Q \\
(x := e)^d & \equiv x := e \\
?Q^d & \equiv ?Q
\end{align*}
\]
Definable Game Operators

Diamond Angel Ops

- \( \cup \) choice
- \( \ast \) repeat
- \( x' = f(x) \) evolve
- \( ?Q \) challenge

Diamond Demon Ops

- \( \cap \) choice
- \( \times \) repeat
- \( x' = f(x)^d \) evolve
- \( ?Q^d \) challenge

\[\text{if}(Q) \alpha \text{ else } \beta \equiv (?Q; \alpha) \cup (?\neg Q; \beta)\]
\[\text{while}(Q) \alpha \equiv (?Q; \alpha)^\ast; ?\neg Q\]
\[\alpha \cap \beta \equiv (\alpha^d \cup \beta^d)^d\]
\[\alpha^\times \equiv ((\alpha^d)^*)^d\]
\[(x' = f(x) \& Q)^d \neq x' = f(x) \& Q\]
\[(x := e)^d \equiv x := e\]
\[?Q^d \neq ?Q\]
Definable Game Operators

Diamond Angel Ops

- ∪: choice
- *: repeat
- \(x' = f(x)\): evolve
- ?Q: challenge

Diamond Demon Ops

- ∩: choice
- ×: repeat
- \(x' = f(x)^d\): evolve
- ?Q^d: challenge

if(Q) α else β ≡ (?Q; α) ∪ (?¬Q; β)

while(Q) α ≡ (?Q; α)^*; ?¬Q

\(α \cap β ≡ (α^d \cup β^d)^d\)

\(α^× ≡ ((α^d)^*)^d\)

\((x' = f(x) \& Q)^d \neq x' = f(x) \& Q\)

\((x := e)^d ≡ x := e\)

?Q^d ↗ ?Q
Simple Examples: EVE and WALL-E

\[(w - e)^2 \leq 1 \land v = f \rightarrow \]
\[\langle ((u := 1 \cap u := -1); \]
\[(g := 1 \cup g := -1); \]
\[t := 0; \]
\[(w' = v, v' = u, e' = f, f' = g, t' = 1 \land t \leq 1)^d \]
\[\times \rangle (w - e)^2 \leq 1 \]

EVE at \(e\) plays Angel’s part controlling \(g\)
WALL-E at \(w\) plays Demon’s part controlling \(u\)
Simple Examples: EVE and WALL·E and the World

\[(w - e)^2 \leq 1 \land v = f \rightarrow \langle ((u := 1 \cap u := -1); \quad (g := 1 \cup g := -1); \quad t := 0; \quad (w' = v, v' = u, e' = f, f' = g, t' = 1 \& t \leq 1)^d \rangle^\times (w - e)^2 \leq 1\]

EVE at e plays Angel’s part controlling g
WALL·E at w plays Demon’s part controlling u
EVE assigned environment’s time to WALL·E
Simple Examples: WALL·E and EVE

\[ (w - e)^2 \leq 1 \land v = f \rightarrow \]
\[ \left[ ((u := 1 \land u := -1); \right. \]
\[ (g := 1 \lor g := -1); \]
\[ t := 0; \]
\[ (w' = v, v' = u, e' = f, f' = g, t' = 1 \& t \leq 1) \]
\[ \times) \right] (w - e)^2 > 1 \]

WALL·E at \( w \) plays Demon’s part controlling \( u \)
EVE at \( e \) plays Angel’s part controlling \( g \)
WALL·E assigned environment’s time to EVE
Definition (Hybrid game $\alpha$: operational semantics)

$\omega_x := e$

$\omega \equiv [e] \omega$

$\omega_x \equiv \omega$
Definition (Hybrid game $\alpha$: operational semantics)

$$x' = f(x) \& Q$$
Definition (Hybrid game $\alpha$: operational semantics)
\[ \omega \vdash e \quad \omega \vdash x \quad \omega \vdash x' = f(x) & Q \]

\[ \omega \vdash \text{repeat} \quad \omega \vdash \text{stop} \]

André Platzer (CMU)
Definition (Hybrid game $\alpha$: operational semantics)

$\omega \equiv \langle \omega \rangle$

$\alpha \equiv \langle \alpha \rangle$

$\alpha; \beta = \langle \alpha \rangle \Rightarrow \langle \beta \rangle$

$\tau \equiv \langle \tau \rangle$

$\tau; \beta = \langle \tau \rangle \Rightarrow \langle \beta \rangle$

$\tau \Rightarrow \sigma = \langle \tau \rangle \Rightarrow \langle \sigma \rangle$

André Platzer (CMU)
Definition (Hybrid game $\alpha$: operational semantics)

$\omega := e \omega$

$\omega x := e \omega x' = f(x) \land Q$
Definition (Hybrid game $\alpha$: operational semantics)
Filibusters

\( \langle x := 0 \cap x := 1 \rangle^* x = 0 \)
Filibusters & The Significance of Finitude

\[ \langle (x := 0 \cap x := 1)^* \rangle x = 0 \]

\( wfd \leadsto \text{false unless } x = 0 \)
Filbusters & The Significance of Finitude

\[\langle (x' = 1^d; x := 0)^* \rangle x = 0\]

\[\langle (x := 0; x' = 1^d)^* \rangle x = 0\]

\[\langle (x := 0 \cap x := 1)^* \rangle x = 0\]

\[\text{wfd} \ni \text{false unless } x = 0\]
\[ \langle (x' = 1^d; x := 0)^* \rangle x = 0 \]

\[ \langle (x := 0; x' = 1^d)^* \rangle x = 0 \]

\[ \langle (x := 0 \land x := 1)^* \rangle x = 0 \]

\[ \text{wfd} \quad \leadsto \text{false unless } x = 0 \]
Filibusters & The Significance of Finitude

\[ \langle x' = 1^d; x := 0 \rangle^* x = 0 \]

\[ \langle x := 0; x' = 1^d \rangle^* x = 0 \]

\[ \langle x := 0 \cap x := 1 \rangle^* x = 0 \]

wfd \[ \leadsto \text{false unless } x = 0 \]

Well-defined games can’t be postponed forever
Outline

1 Learning Objectives

2 Motivation

3 Hybrid Games
   - Choices & Nondeterminism
   - Control & Dual Control
   - Hybrid Games
   - Differential Game Logic
   - Demon’s Controls
   - Operational Game Semantics
   - Filibusters & Finitude

4 Example: Robot Factory

5 Summary
Example: Robot Factory Decentralized Automation

Model
- \((x, y)\) robot coordinates
- \((v_x, v_y)\) velocities
- conveyor belts may instantaneously increase robot’s velocity by \((c_x, c_y)\)

Primary objectives of the robot
- Leave within time \(\varepsilon\)
- Never leave outer

Challenges
- Distributed, physical environment
- Possibly conflicting secondary objectives
Example (Robot-Demon vs. Angel-Factory Environment)

\[ \left( ?\text{true} \cup (? (x < e_x \land y < e_y \land eff_1 = 1); \ v_x := v_x + c_x; \ eff_1 := 0) \right) \]

\[ \cup (? (e_x \leq x \land y \leq f_y \land eff_2 = 1); \ v_y := v_y + c_y; \ eff_2 := 0) \]
Example (Robot-Demon vs. Angel-Factory Environment)

\[
\begin{align*}
\left( \text{true} \cup (\text{x} < e_x \land y < e_y \land \text{eff}_1 = 1); \ v_x := v_x + c_x; \ \text{eff}_1 := 0 \right) & \quad \text{// belt} \\
\cup (e_x \leq x \land y \leq f_y \land \text{eff}_2 = 1); \ v_y := v_y + c_y; \ \text{eff}_2 := 0 \right) \\
(a_x := \ast; \ ?(-A \leq a_x \leq A); \\
a_y := \ast; \ ?(-A \leq a_y \leq A); & \quad \text{// “independent” robot acceleration} \\
t_s := 0 \right)^d \\
\end{align*}
\]
Example (Robot-Demon vs. Angel-Factory Environment)

\[
\left( \text{true} \cup \left( x < e_x \land y < e_y \land e_{f_1} = 1 \right) ; v_x := v_x + c_x ; e_{f_1} := 0 \right) \quad \text{// belt}
\]

\[
\cup \left( e_x \leq x \land y \leq f_y \land e_{f_2} = 1 \right) ; v_y := v_y + c_y ; e_{f_2} := 0 \right) ;
\]

\[
( a_x := * ; ?(-A \leq a_x \leq A) ;
\]

\[
a_y := * ; ?(-A \leq a_y \leq A) ; \quad \text{// “independent” robot acceleration}
\]

\[
t_s := 0 \right)^d ;
\]

\[
( x' = v_x , y' = v_y , v'_x = a_x , v'_y = a_y , t' = 1 , t'_s = 1 \land t_s \leq \varepsilon ) ;
\]
Example (Robot-Demon vs. Angel-Factory Environment)

\[
\left( (\text{true} \cup (? (x < e_x \land y < e_y \land \text{eff}_1 = 1)); \quad v_x := v_x + c_x; \quad \text{eff}_1 := 0) \right)
\]

\[
\cup (?(e_x \leq x \land y \leq f_y \land \text{eff}_2 = 1); \quad v_y := v_y + c_y; \quad \text{eff}_2 := 0) \right);
\]

\[
(a_x := \ast; \quad ?(-A \leq a_x \leq A);
\]

\[
a_y := \ast; \quad ?(-A \leq a_y \leq A); \quad \text{“independent” robot acceleration}
\]

\[
t_s := 0 \right)^d;
\]

\[
\left( (x' = v_x, y' = v_y, v'_x = a_x, v'_y = a_y, t' = 1, t'_s = 1 \land t_s \leq \varepsilon) \right)
\]

\[
\cap (? (a_x v_x \leq 0 \land a_y v_y \leq 0)) \right)^d; \quad \text{brake}
\]

\[
\text{if } v_x = 0 \text{ then } a_x := 0 \text{ fi;} \quad \text{per direction: no time lock}
\]

\[
\text{if } v_y = 0 \text{ then } a_y := 0 \text{ fi;}
\]

\[
(x' = v_x, y' = v_y, v'_x = a_x, v'_y = a_y, t' = 1, t'_s = 1 \land t_s \leq \varepsilon) \right)^*;
\]
Robot Factory Automation (RF)

**Proposition (Robot stays in □)**

\[
\models (x = y = 0 \land v_x = v_y = 0 \land \text{Controllability Assumptions}) \\
\rightarrow [RF](x \in [l_x, r_x] \land y \in [l_y, r_y])
\]

**Proposition (Stays in □ and leaves □ on time)**

\[
[RF|_x]: \text{RF projected to the x-axis}
\]

\[
\models (x = 0 \land v_x = 0 \land \text{Controllability Assumptions}) \\
\rightarrow [RF|_x](x \in [l_x, r_x] \land (t \geq \varepsilon \rightarrow x \geq x_b))
\]

CADE’12
Outline

1. Learning Objectives
2. Motivation
3. Hybrid Games
   - Choices & Nondeterminism
   - Control & Dual Control
   - Hybrid Games
   - Differential Game Logic
   - Demon’s Controls
   - Operational Game Semantics
   - Filibusters & Finitude
4. Example: Robot Factory
5. Summary
Differential Game Logic: Syntax

**Definition (Hybrid game $\alpha$)**

$$x := e \mid ?Q \mid x' = f(x) \mid \alpha \cup \beta \mid \alpha;\beta \mid \alpha^* \mid \alpha^d$$

**Definition (dGL Formula $P$)**

$$p(e_1, \ldots, e_n) \mid e \geq \bar{e} \mid \neg P \mid P \land Q \mid \forall x \ P \mid \exists x \ P \mid \langle \alpha \rangle P \mid [\alpha] P$$

Discrete Assign Test Game Differential Equation Choice Game Seq. Game Repeat Game Dual Game

All Reals Some Reals Angel Wins Demon Wins

TOCL'15
Summary

- Differential game logic
- Logic for hybrid games
- Compositional PL + logic
- Discrete + continuous + adversarial
- Operational semantics (informally)

Next lecture

1. Formal semantics
André Platzer.
Foundations of cyber-physical systems.

André Platzer.
Differential game logic.

André Platzer.
Logics of dynamical systems.
In LICS [10], pages 13–24.

André Platzer.
Differential dynamic logic for hybrid systems.
A uniform substitution calculus for differential dynamic logic.
doi:10.1007/978-3-319-21401-6_32.

André Platzer.
The complete proof theory of hybrid systems.
In LICS [10], pages 541–550.
doi:10.1109/LICS.2012.64.

André Platzer.
A complete axiomatization of quantified differential dynamic logic for distributed hybrid systems.
Special issue for selected papers from CSL’10.

André Platzer.
Stochastic differential dynamic logic for stochastic hybrid programs.
Jan-David Quesel and André Platzer.
Playing hybrid games with KeYmaera.
doi:10.1007/978-3-642-31365-3_34.