

Interpretable Strategy Synthesis for Competitive Games

Oral Prelim Presentation

Abhijeet Krishnan

Department of Computer Science
North Carolina State University

April 23, 2023

Previous Work

- Krishnan, Abhijeet and Chris Martens. “Synthesizing Chess Tactics from Player Games.” In *Workshop on Artificial Intelligence for Strategy Games (SG) and Esports Analytics (EA), 18th AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment*. 2022 (in press).
- Krishnan, Abhijeet and Chris Martens. “Towards the Automatic Synthesis of Interpretable Chess Tactics.” In *Explainable Agency in Artificial Intelligence Workshop, 36th AAAI Conference on Artificial Intelligence*. 2022.
- Krishnan, Abhijeet, Aaron Williams, and Chris Martens. “Towards Action Model Learning for Player Modeling.” *Proceedings of the AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment*. Vol. 16. No. 1. 2020.
- Krishnan, Abhijeet and Chris Martens. “Rule-based Cognitive Modeling via Human-Computer Interaction”. Poster presented at: *5th LAS Research Symposium*; 2019 Dec 10; Raleigh, NC.

Story Time!



Yogender Pal

Figure 1: Priya, a normal girl

Story Time!



Netflix

Story Time!

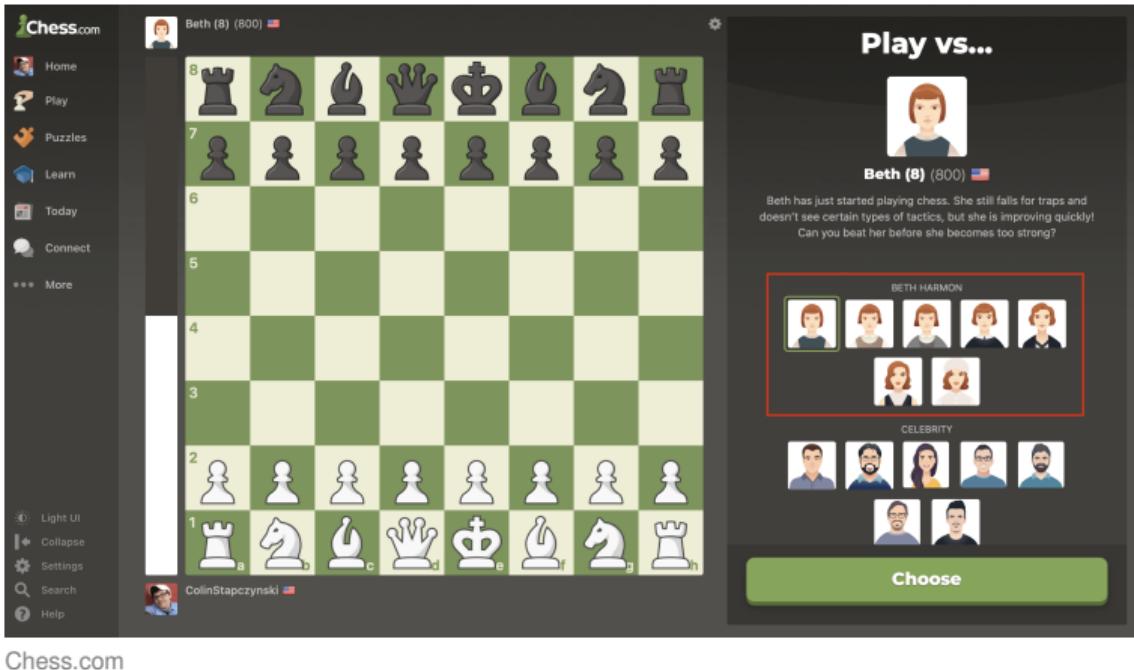


Figure 2: Beth Harmon bots on Chess.com

Story Time!



Chess.com

Figure 3: Beth Harmon (bot) at 8 years old

Story Time!



ChessKid



ChessKid

Story Time!



Chess.com

Figure 4: Beth Harmon (bot) at 15 years old

Story Time!



Arjun Somasekharan

Figure 5: What should Priya do now?

Story Time!

Could the Beth Harmon bots *explain* their *strategy* to Priya to help her get better?

Could we *synthesize* good strategies for helping people get better at games?

Strategy

Observation

Players tend to identify *strategies* when trying to find ways to win in a competitive game.

Strategy

Observation

Players tend to identify *strategies* when trying to find ways to win in a competitive game.

- Can be *communicated* to another player

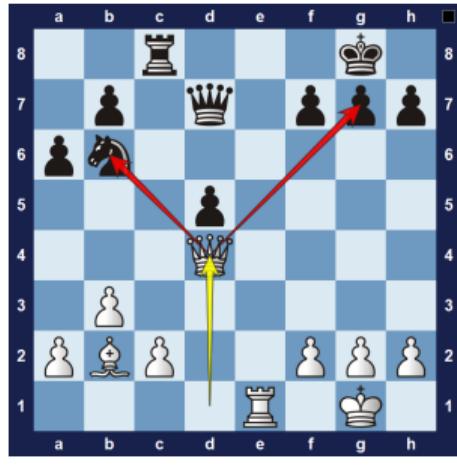
Strategy

Observation

Players tend to identify *strategies* when trying to find ways to win in a competitive game.

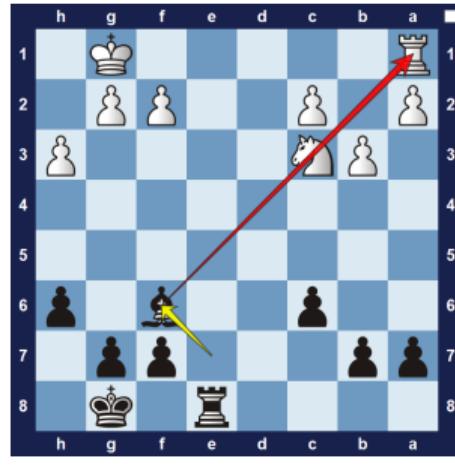
- Can be communicated to another player
- Help a player *decide* which actions to take in-game

Examples



Chessfox

Figure 6: An example of the *fork* tactic in chess



Chessfox

Figure 7: An example of the *pin* tactic in chess

Examples



Go Full Build

Figure 8: A *cannon rush* in progress against a Terran opponent in the game *StarCraft II*

Why is this useful?

- Esports is a *massive* industry

Why is this useful?

- Esports is a *massive* industry
- Could be used to *coach players* at all levels of skill

Why is this useful?

- Esports is a *massive* industry
- Could be used to *coach players* at all levels of skill
 - Over 200,000 active ChessKid users

Why is this useful?

- Esports is a *massive* industry
- Could be used to *coach players* at all levels of skill
 - Over 200,000 active ChessKid users
- Better strategies → higher player skill → *more earning* potential

Tournament	Game	Prize Pool (USD)
World Blitz Chess Championship	Chess	350,000
IEM Katowice	StarCraft II	500,000

Why is this useful?

- Esports is a *massive* industry
- Could be used to *coach players* at all levels of skill
 - Over 200,000 active ChessKid users
- Better strategies → higher player skill → *more earning* potential

Tournament	Game	Prize Pool (USD)
World Blitz Chess Championship	Chess	350,000
IEM Katowice	StarCraft II	500,000

- Could be used to *explain* artificial agent decisions

Thesis Statement

Thesis Statement

A *computational model* of a game strategy, along with a *learning method*, could meet the goals of discovering good, communicable strategies and impact the fields of competitive esports and explainable AI.

Summary

Research Thrust	RQ	Sub-RQ	Publication
ISS Framework	RQ1	–	EAAI '22 (Krishnan and Martens 2022b)
		RQ2(a)	EAAI '22 (Krishnan and Martens 2022b)
ISS for Chess	RQ2	RQ2(b)	SG+EA Workshop @ AIIDE '22 (Krishnan and Martens 2022a)
		RQ2(c)	(<i>under review</i>) (Krishnan, Martens, and Jhala 2023)
ISS for MicroRTS	RQ3	RQ3(a)	<i>Proposed Work</i> (ACG 2023)
		RQ3(b)	<i>Proposed Work</i> (AAMAS 2024)

RQs

RQs

RQ1

How do we formally define the problem of *Interpretable Strategy Synthesis* (ISS)?

RQs

RQ1

How do we formally define the problem of *Interpretable Strategy Synthesis* (ISS)?

RQ2

How do we approach the problem of ISS for the game of *chess*?

RQs

RQ1

How do we formally define the problem of *Interpretable Strategy Synthesis* (ISS)?

RQ2

How do we approach the problem of ISS for the game of *chess*?

RQ3

How do we approach the problem of ISS for the game of *MicroRTS*?

ISS Framework

RQ1

How do we formally define the problem of *Interpretable Strategy Synthesis* (ISS)?

Elements of a Good Framework

Elements of a Good Framework

- Facilitates comparison between *multiple algorithms*

Elements of a Good Framework

- Facilitates comparison between multiple algorithms
- Facilitates comparison between *multiple strategy representations*

Elements of a Good Framework

- Facilitates comparison between multiple algorithms
- Facilitates comparison between multiple strategy representations
- Facilitates comparison across *multiple games*

Elements of a Good Framework

- Facilitates comparison between multiple algorithms
- Facilitates comparison between multiple strategy representations
- Facilitates comparison across multiple games
- Provides a clear *definition* of interpretability

The Need for a Framework

Paper	Number Used			Interpretability
	Domains	Models	Algorithms	
Spronck, Sprinkhuizen-Kuyper, and Postma (2004)	2	1	1	✗
Mesentier Silva et al. (2016)	1	1	4	✓
Butler, Torlak, and Popović (2017)	1	1	1	✗
Canaan et al. (2018)	1	1	1	✗
Freitas, Souza, and Bernardino (2018)	1	1	1	✗
Mariño et al. (2021)	1	1	1	✗
Krishnan and Martens (2022a)	1	1	1	✗
Mariño and Toledo (2022)	1	1	1	✗
Medeiros, Aleixo, and Lelis (2022)	2	1	2	✗

Table 1: List of works in ISS

Interpretable Strategy Synthesis (ISS)

Definition (ISS)

Given a —

- Game environment \mathcal{G}
- Strategy model \mathcal{M}
- Performance measure $\mathcal{R}: \mathcal{M} \rightarrow \mathbb{R}$
- Interpretability measure $\mathcal{I}: \mathcal{M} \rightarrow \mathbb{R}$

The problem of ISS is to find a strategy σ^* s.t. —

$$\sigma^* \doteq \arg \max_{\sigma} \mathcal{R}(\sigma) \mathcal{I}(\sigma), \sigma \in \mathcal{M}$$

Strategy (σ)

- Strategy = RL policy – universal applicability

Strategy (σ)

- Strategy = RL policy – universal applicability
- Strategy *not* applicable to all states

Strategy (σ)

- Strategy = RL policy – universal applicability
- Strategy *not* applicable to all states
- Describes an *oft-seen pattern* in gameplay

Strategy (σ)

- Strategy = RL policy – universal applicability
- Strategy *not* applicable to all states
- Describes an *oft-seen pattern* in gameplay

• Formal Definition

Strategy Model (\mathcal{M})

- Defines the *space* of strategies

Strategy Model (\mathcal{M})

- Defines the *space* of strategies
- Examples —

Strategy Model (\mathcal{M})

- Defines the *space* of strategies
- Examples —
 - if-then rules

Strategy Model (\mathcal{M})

- Defines the *space* of strategies
- Examples —
 - if-then rules
 - decision trees

Strategy Model (\mathcal{M})

- Defines the *space* of strategies
- Examples —
 - if-then rules
 - decision trees
 - programmatic scripts

Performance Measure ($\mathcal{R}(\sigma)$)

- How *good* a strategy is

Performance Measure ($\mathcal{R}(\sigma)$)

- How *good* a strategy is
- Players generally study good strategies

Performance Measure ($\mathcal{R}(\sigma)$)

- How *good* a strategy is
- Players generally study good strategies
- Examples —

Performance Measure ($\mathcal{R}(\sigma)$)

- How *good* a strategy is
- Players generally study good strategies
- Examples —
 - win rate

Performance Measure ($\mathcal{R}(\sigma)$)

- How *good* a strategy is
- Players generally study good strategies
- Examples —
 - win rate
 - material advantage (chess)

Performance Measure ($\mathcal{R}(\sigma)$)

- How *good* a strategy is
- Players generally study good strategies
- Examples —
 - win rate
 - material advantage (chess)
 - resources harvested (MicroRTS)

Interpretability Measure ($\mathcal{I}(\sigma)$)

- How *interpretable* a strategy is

Interpretability Measure ($\mathcal{I}(\sigma)$)

- How *interpretable* a strategy is
- Players need to be able to *understand* a strategy to benefit from it

Interpretability Measure ($\mathcal{I}(\sigma)$)

- How *interpretable* a strategy is
- Players need to be able to *understand* a strategy to benefit from it
- Examples —

Interpretability Measure ($\mathcal{I}(\sigma)$)

- How *interpretable* a strategy is
- Players need to be able to *understand* a strategy to benefit from it
- Examples —
 - number of statements (programmatic script)

Interpretability Measure ($\mathcal{I}(\sigma)$)

- How *interpretable* a strategy is
- Players need to be able to *understand* a strategy to benefit from it
- Examples —
 - number of statements (programmatic script)
 - number of nodes (decision tree)

Interpretability Measure ($\mathcal{I}(\sigma)$)

- How *interpretable* a strategy is
- Players need to be able to *understand* a strategy to benefit from it
- Examples —
 - number of statements (programmatic script)
 - number of nodes (decision tree)
 - set of conditions and actions used (if-then rule)

Interpretability Measure ($\mathcal{I}(\sigma)$)

- How *interpretable* a strategy is
- Players need to be able to *understand* a strategy to benefit from it
- Examples —
 - number of statements (programmatic script)
 - number of nodes (decision tree)
 - set of conditions and actions used (if-then rule)
 - visualization method (text-based output vs. diagrams)

Interpretability Measure ($\mathcal{I}(\sigma)$)

- How *interpretable* a strategy is
- Players need to be able to *understand* a strategy to benefit from it
- Examples —
 - number of statements (programmatic script)
 - number of nodes (decision tree)
 - set of conditions and actions used (if-then rule)
 - visualization method (text-based output vs. diagrams)
- Mainly considering strategy models with *text-based* representations

Interpretable Strategy Synthesis (ISS)

Definition (ISS)

Given a —

- Game environment \mathcal{G}
- Strategy model \mathcal{M}
- Performance measure $\mathcal{R}: \mathcal{M} \rightarrow \mathbb{R}$
- Interpretability measure $\mathcal{I}: \mathcal{M} \rightarrow \mathbb{R}$

The problem of ISS is to find a strategy σ^* s.t. —

$$\sigma^* \doteq \arg \max_{\sigma} \mathcal{R}(\sigma) \mathcal{I}(\sigma), \sigma \in \mathcal{M}$$

ISS for Chess

RQ2

How do we approach the problem of Interpretable Strategy Synthesis for the game of *chess*?

Why Chess?

- Popular game with a *long* competitive history

Why Chess?

- *Popular* game with a *long* competitive history
- Has a large number of player-discovered strategies

Why Chess?

- Popular game with a *long* competitive history
- Has a large number of *player-discovered strategies*
- Extensive use as a testbed for AI

Why Chess?

- Popular game with a *long* competitive history
- Has a large number of player-discovered strategies
- Extensive use as a *testbed for AI*

Towards ISS for Chess

- Strategy model for chess
- Performance measure for chess
- Interpretability measure for chess

Towards ISS for Chess

- Strategy model for chess
- Performance measure for chess
- Interpretability measure for chess

RQ2(a)

Could we represent known chess tactics as a strategy model for chess and develop metrics to show that they suggest better moves than a random baseline?

Towards ISS for Chess

- Strategy model for chess
- Performance measure for chess
- Interpretability measure for chess

RQ2(a)

Could we represent known chess tactics as a *strategy model* for chess and develop metrics to show that they suggest better moves than a random baseline?

Strategy Model for Chess

First-Order (FO) Logic Rule

Strategy Model for Chess

First-Order (FO) Logic Rule

Predicate Vocabulary

Strategy Model for Chess

First-Order (FO) Logic Rule

Predicate Vocabulary

```
tactic(Position, Move) ←  
    feature_1(…),  
    feature_2(…),  
    :  
    feature_n(…)
```

Figure 9: Our chess strategy model expressed in Prolog pseudocode

Strategy Model for Chess

First-Order (FO) Logic Rule

```
tactic(Position, Move) ←  
    feature_1(…),  
    feature_2(…),  
    :  
    feature_n(…)
```

Predicate Vocabulary

- **Position** =

- [contents(c2,pawn,white),
 contents(g8,knight,black),
 contents(e8,king,black),
 turn(white),kingside_castle(white),...]

Figure 9: Our chess strategy model expressed in Prolog pseudocode

Strategy Model for Chess

First-Order (FO) Logic Rule

```
tactic(Position, Move) ←  
    feature_1(…),  
    feature_2(…),  
    :  
    feature_n(…)
```

Predicate Vocabulary

- **Position** =
[contents(c2,pawn,white),
 contents(g8,knight,black),
 contents(e8,king,black),
 turn(white),kingside_castle(white),...]
- **Move** = [a7, a8, queen]

Figure 9: Our chess strategy model expressed in Prolog pseudocode

Strategy Model for Chess

First-Order (FO) Logic Rule

```
tactic(Position, Move) ←  
    feature_1(...),  
    feature_2(...),  
    :  
    feature_n(...)
```

Figure 9: Our chess strategy model expressed in Prolog pseudocode

Predicate Vocabulary

- **Position** =
[contents(c2,pawn,white),
 contents(g8,knight,black),
 contents(e8,king,black),
 turn(white),kingside_castle(white),...]
- **Move** = [a7, a8, queen]
- **Features** =
 - attacks(Pos, Sq1, Sq2)
 - in_check(Pos, Side)
 - is_empty(Pos, Squares)

Example

```
fork(Position,Move) ←  
    legal_move(Position,Move),  
    move(Move,_,To,_),  
    make_move(Position,Move>NewPosition),  
    can_capture(NewPosition,To,ForkSquare1),  
    can_capture(NewPosition,To,ForkSquare2),  
    different(ForkSquare1,ForkSquare2).
```

Figure 10: An interpretation of the *fork* tactic from the chess literature using our predicate vocabulary.

Example

```
fork(Position,Move) ←  
    legal_move(Position,Move),  
    move(Move,_,To,_),  
    make_move(Position,Move,NewPosition),  
    can_capture(NewPosition,To,ForkSquare1),  
    can_capture(NewPosition,To,ForkSquare2),  
    different(ForkSquare1,ForkSquare2).
```

Figure 10: An interpretation of the *fork* tactic from the chess literature using our predicate vocabulary.

Towards ISS for Chess

- Strategy model for chess
- Performance measure for chess
- Interpretability measure for chess

RQ2(a)

Could we represent known chess tactics as a strategy model for chess and develop metrics to show that they suggest better moves than a random baseline?

Towards ISS for Chess

- Strategy model for chess
- *Performance measure* for chess
- *Interpretability measure* for chess

RQ2(a)

Could we represent known chess tactics as a strategy model for chess and develop *metrics* to show that they suggest better moves than a random baseline?

Performance Measure

Divergence

Performance Measure

Divergence

- How *different* is one strategy from another?

Performance Measure

Divergence

- How *different* is one strategy from another?
- High divergence → strategies are very different

Performance Measure

Divergence

- How *different* is one strategy from another?
- High divergence → strategies are very different
- Low divergence → strategies are quite similar

Performance Measure

Divergence

- How *different* is one strategy from another?
- High divergence → strategies are very different
- Low divergence → strategies are quite similar
- Difference in terms of *perceived evaluation* of moves

Performance Measure

Divergence

- How *different* is one strategy from another?
- High divergence → strategies are very different
- Low divergence → strategies are quite similar
- Difference in terms of *perceived evaluation* of moves
- Who is “perceiving”?

Performance Measure

Divergence

- How *different* is one strategy from another?
- High divergence → strategies are very different
- Low divergence → strategies are quite similar
- Difference in terms of *perceived evaluation* of moves
- Who is “perceiving”?
 - Chess-playing agents with an *evaluation function* (chess “engines”)

Performance Measure

Divergence

- How *different* is one strategy from another?
- High divergence → strategies are very different
- Low divergence → strategies are quite similar
- Difference in terms of *perceived evaluation* of moves
- Who is “perceiving”?
 - Chess-playing agents with an *evaluation function* (chess “engines”)
 - e.g., Stockfish 14, Leela Chess Zero

Performance Measure

Divergence

- How *different* is one strategy from another?
- High divergence → strategies are very different
- Low divergence → strategies are quite similar
- Difference in terms of *perceived evaluation* of moves
- Who is “perceiving”?
 - Chess-playing agents with an *evaluation function* (chess “engines”)
 - e.g., Stockfish 14, Leela Chess Zero



Equation

Interpretability Measure

Interpretability Measure

- No explicit interpretability measure! Only qualitative arguments

Interpretability Measure

- No explicit interpretability measure! Only qualitative arguments
- Human players *think* and *train* using chess tactics (Szabo 1984; Gobet and Jansen 2006)

Interpretability Measure

- No explicit interpretability measure! Only qualitative arguments
- Human players *think* and *train* using chess tactics (Szabo 1984; Gobet and Jansen 2006)
- FO-logic used extensively to model chess patterns (Berliner 1975; Pitrat 1977; Wilkins 1979; Huberman 1968; Bramer 1977; Bratko 1982; Morales 1992)

Interpretability Measure

- No explicit interpretability measure! Only qualitative arguments
- Human players *think* and *train* using chess tactics (Szabo 1984; Gobet and Jansen 2006)
- FO-logic used extensively to model chess patterns (Berliner 1975; Pitrat 1977; Wilkins 1979; Huberman 1968; Bramer 1977; Bratko 1982; Morales 1992)
- Logic rules are *acknowledged to be interpretable* (Zhang et al. 2021)

Towards ISS for Chess

- Strategy model for chess
- Performance measure for chess
- Interpretability measure for chess

RQ2(a)

Could we represent known chess tactics as a strategy model for chess and develop metrics to *show that they suggest better moves than a random baseline?*

Evaluating Chess Tactics¹

¹Krishnan and Martens 2022b.

Evaluating Chess Tactics¹

- PAL (Morales 1992) $\xrightarrow{\text{learn}}$ *known* chess patterns (tactics) PAL

¹Krishnan and Martens 2022b.

Evaluating Chess Tactics¹

- PAL (Morales 1992) $\xrightarrow{\text{learn}}$ known chess patterns (tactics) PAL
- tactics $\xrightarrow{\text{translate}}$ chess strategy model

¹Krishnan and Martens 2022b.

Evaluating Chess Tactics¹

- PAL (Morales 1992) $\xrightarrow{\text{learn}}$ known chess patterns (tactics) PAL
- tactics $\xrightarrow{\text{translate}}$ chess strategy model
- Divergence(chess strategies, *human beginner* games)

¹Krishnan and Martens 2022b.

Evaluating Chess Tactics¹

- PAL (Morales 1992) $\xrightarrow{\text{learn}}$ known chess patterns (tactics) PAL
- tactics $\xrightarrow{\text{translate}}$ chess strategy model
- Divergence(chess strategies, human beginner games)
- Divergence(*random baseline*, human beginner games)

¹Krishnan and Martens 2022b.

Evaluating Chess Tactics¹

- PAL (Morales 1992) $\xrightarrow{\text{learn}}$ known chess patterns (tactics) PAL
- tactics $\xrightarrow{\text{translate}}$ chess strategy model
- Divergence(chess strategies, human beginner games)
- Divergence(random baseline, human beginner games)
- Both using strong/weak engine

¹Krishnan and Martens 2022b.

Results

Tactic	Divergence	
	Strong	Weak
can_threat	378.94	9.22
can_check	549.19	4.02
can_fork	676.45	4.67
discovered_check	338.55	18.64
discovered_threat	375.97	1.19
skewer	748.40	5.41
pin	526.45	4.90
random	328.09	8.28

Table 2: Divergence for each tactic

Analysis

- *Higher than random* divergence from human beginners (strong engine)

Analysis

- *Higher than random* divergence from human beginners (strong engine)
- *Lower than random* divergence from human beginners (weak engine)

Analysis

- *Higher than random* divergence from human beginners (strong engine)
- *Lower than random* divergence from human beginners (weak engine)
- Known chess strategies approximate human beginners better than random according to a weak engine

Learning Chess Strategy Models

- Strategy model for chess
- Performance measure for chess
- Interpretability measure for chess

RQ2(b)

Do the chess strategies learned using inductive logic programming outperform a random baseline in how closely their divergence scores approximate a beginner player?

Learning Chess Strategy Models

- Strategy model for chess
- Performance measure for chess
- Interpretability measure for chess
- Learning algorithm for chess strategies

RQ2(b)

Do the chess strategies learned using inductive logic programming outperform a random baseline in how closely their divergence scores approximate a beginner player?

Learning Chess Strategies using ILP²

²Krishnan and Martens 2022a.

Learning Chess Strategies using ILP²

- Inductive Logic Programming (ILP): *symbolic ML* technique ILP

²Krishnan and Martens 2022a.

Learning Chess Strategies using ILP²

- Inductive Logic Programming (ILP): *symbolic ML* technique ILP
- ISS for chess $\langle \mathcal{G}, \mathcal{M}, \mathcal{R} \rangle \xrightarrow{\text{translate}} \text{ILP problem } \langle E^+, E^-, B \rangle$

²Krishnan and Martens 2022a.

Learning Chess Strategies using ILP²

- Inductive Logic Programming (ILP): *symbolic ML* technique ILP
- ISS for chess $\langle \mathcal{G}, \mathcal{M}, \mathcal{R} \rangle \xrightarrow{\text{translate}} \text{ILP problem } \langle E^+, E^-, B \rangle$
- ILP system($\langle E^+, E^-, B \rangle$) $\xrightarrow{\text{learn}}$ chess strategies

²Krishnan and Martens 2022a.

Learning Chess Strategies using ILP²

- Inductive Logic Programming (ILP): *symbolic ML* technique ILP
- ISS for chess $\langle \mathcal{G}, \mathcal{M}, \mathcal{R} \rangle \xrightarrow{\text{translate}} \text{ILP problem } \langle E^+, E^-, B \rangle$
- ILP system($\langle E^+, E^-, B \rangle$) $\xrightarrow{\text{learn}}$ chess strategies
- Use *divergence* to evaluate learned chess strategies

²Krishnan and Martens 2022a.

Learning Chess Strategies using ILP²

- Inductive Logic Programming (ILP): *symbolic ML* technique ILP
- ISS for chess $\langle \mathcal{G}, \mathcal{M}, \mathcal{R} \rangle \xrightarrow{\text{translate}} \text{ILP problem } \langle E^+, E^-, B \rangle$
- ILP system($\langle E^+, E^-, B \rangle$) $\xrightarrow{\text{learn}}$ chess strategies
- Use *divergence* to evaluate learned chess strategies
- Compare to random, strong/weak engine baselines

²Krishnan and Martens 2022a.

Results

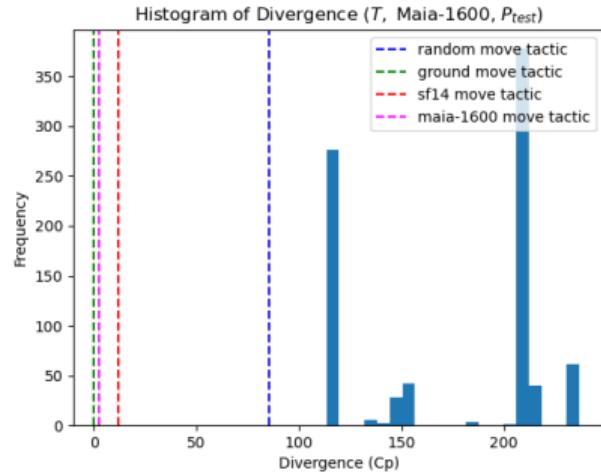


Figure 11: Divergence histogram for T evaluated using *weak* engine

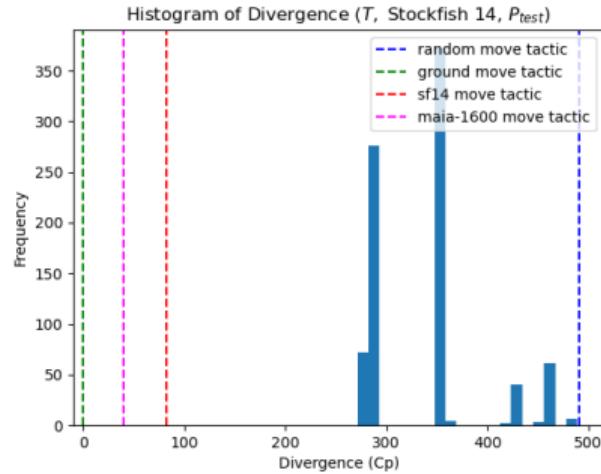


Figure 12: Divergence histogram for T evaluated using *strong* engine

Analysis

- *Lower than random* divergence from human beginners (strong engine)

Analysis

- *Lower than random* divergence from human beginners (strong engine)
- *Higher than random* divergence from human beginners (weak engine)

Analysis

- *Lower than random* divergence from human beginners (strong engine)
- *Higher than random* divergence from human beginners (weak engine)
- Learned chess strategies approximate human beginners better than random according to a strong engine

Improving the ILP Learning Method

- How do we *improve* upon “better than random”?

Improving the ILP Learning Method

- How do we *improve* upon “better than random”?

RQ2(c)

Do the chess strategies learned by an ILP system incorporating the changes of the new predicate vocabulary and precision/recall-based constraints produce moves better than those learned by an ILP system without these modifications?

Improvements using Precision/Recall-based Constraints³

³Krishnan, Martens, and Jhala 2023.

Improvements using Precision/Recall-based Constraints³

- *Limit* chess strategy search space using precision/recall constraints

Constraints

³Krishnan, Martens, and Jhala 2023.

Improvements using Precision/Recall-based Constraints³

- *Limit* chess strategy search space using precision/recall constraints Constraints
- Introduce a *new* predicate vocabulary Vocabulary

³Krishnan, Martens, and Jhala 2023.

Improvements using Precision/Recall-based Constraints³

- *Limit* chess strategy search space using precision/recall constraints Constraints
- Introduce a *new* predicate vocabulary Vocabulary
- Conduct *ablateive study* to measure impact of contributions

³Krishnan, Martens, and Jhala 2023.

Improvements using Precision/Recall-based Constraints³

- *Limit* chess strategy search space using precision/recall constraints Constraints
- Introduce a *new* predicate vocabulary Vocabulary
- Conduct *ablative study* to measure impact of contributions
 - Learn strategies using systems with/without constraints

³Krishnan, Martens, and Jhala 2023.

Improvements using Precision/Recall-based Constraints³

- *Limit* chess strategy search space using precision/recall constraints Constraints
- Introduce a *new* predicate vocabulary Vocabulary
- Conduct *ablative study* to measure impact of contributions
 - Learn strategies using systems with/without constraints
 - Measure average strategy divergence

³Krishnan, Martens, and Jhala 2023.

Improvements using Precision/Recall-based Constraints³

- *Limit* chess strategy search space using precision/recall constraints Constraints
- Introduce a *new* predicate vocabulary Vocabulary
- Conduct *ablative study* to measure impact of contributions
 - Learn strategies using systems with/without constraints
 - Measure average strategy divergence
 - Test decrease vs. old system using *one-sided Welch's t-test*

³Krishnan, Martens, and Jhala 2023.

Results

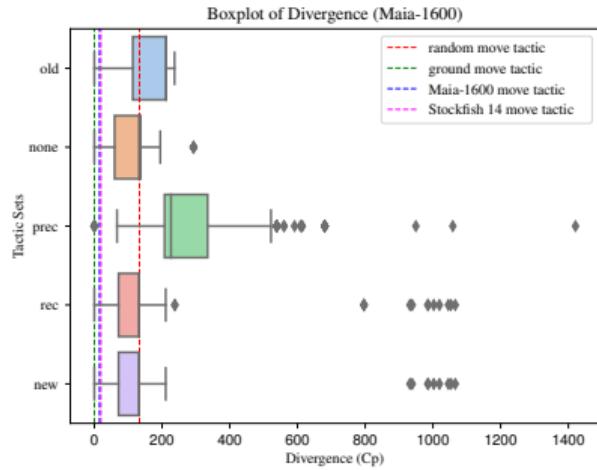


Figure 13: Boxplot of tactic divergence (evaluated using *weak* engine) for each system

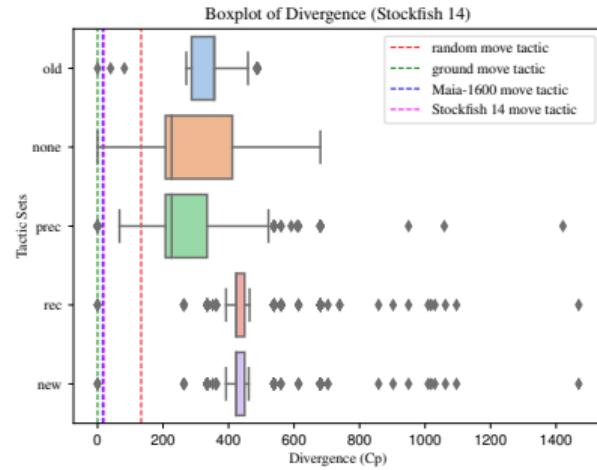


Figure 14: Boxplot of tactic divergence (evaluated using *strong* engine) for each system

Analysis

- New predicate vocabulary → improves divergence! ($p < 0.01$)

Analysis

- New predicate vocabulary → improves divergence! ($p < 0.01$)
- precision constraint → improves divergence *only* when measured using strong engine

Analysis

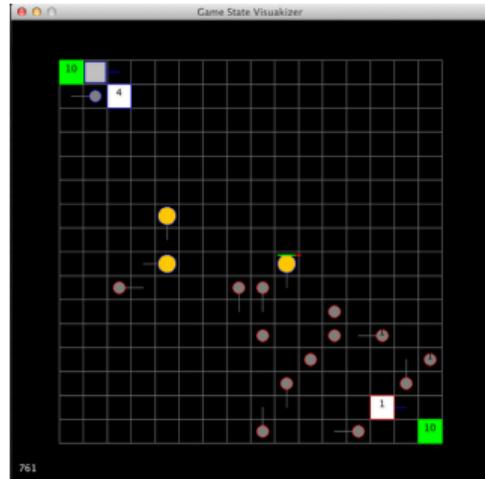
- New predicate vocabulary → improves divergence! ($p < 0.01$)
- precision constraint → improves divergence *only* when measured using strong engine
- recall constraint → improves divergence *only* when measured using weak engine

ISS for MicroRTS

RQ3

How do we approach the problem of Interpretable Strategy Synthesis for the game of *MicroRTS*?

Why MicroRTS?

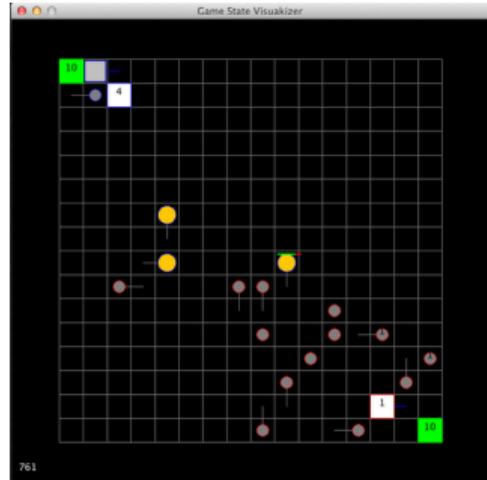


Google Code Archive

Figure 15: A MicroRTS game in progress

Why MicroRTS?

- Simplified **real-time strategy** game for AI research (Ontanon 2021)

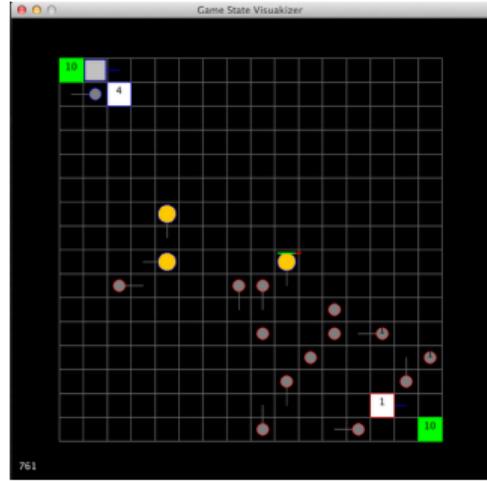


Google Code Archive

Figure 15: A MicroRTS game in progress

Why MicroRTS?

- Simplified real-time strategy game for AI research (Ontanon 2021)
- Popular genre for esport titles

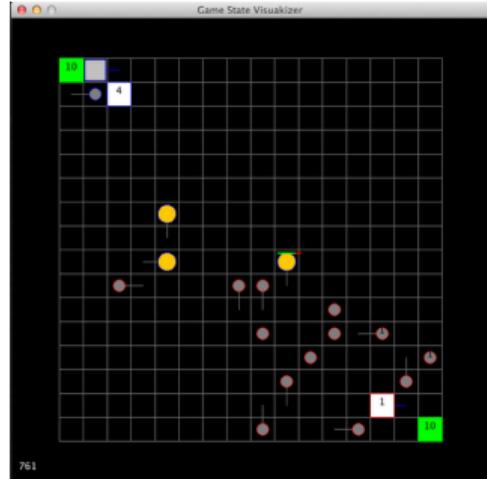


Google Code Archive

Figure 15: A MicroRTS game in progress

Why MicroRTS?

- Simplified real-time strategy game for AI research (Ontanon 2021)
- Popular genre for esport titles
- Active research community

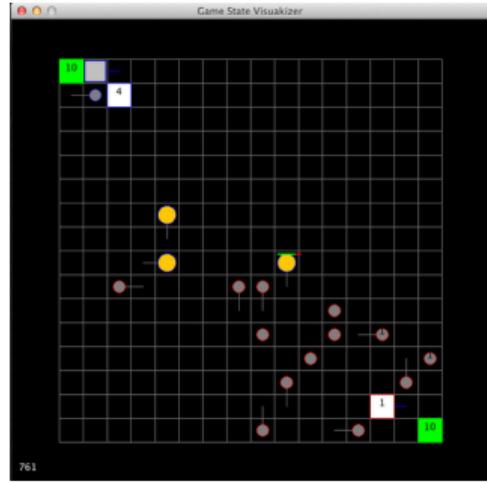


Google Code Archive

Figure 15: A MicroRTS game in progress

Why MicroRTS?

- Simplified real-time strategy game for AI research (Ontanon 2021)
- Popular genre for esport titles
- Active research community
- Qualitatively different from chess – *real-time, partially observable*



Google Code Archive

Figure 15: A MicroRTS game in progress

Towards ISS for MicroRTS

- Strategy model for MicroRTS
- Performance measure for MicroRTS
- Interpretability measure for MicroRTS
- Learning method for MicroRTS strategies

Towards ISS for MicroRTS

- Strategy model for MicroRTS
 - Performance measure for MicroRTS
 - Interpretability measure for MicroRTS
 - Learning method for MicroRTS strategies
- 
- SynProS

SynProS Competition

SynProS Competition

- SynProS: **Syn**thesis of
Programmatic **S**trategies

SynProS Competition

- SynProS: **Syn**thesis of **P**rogrammatic **S**trategies
- Research competition (Moraes 2021) to test ISS approaches for MicroRTS with a *fixed strategy model*

SynProS Competition

- SynProS: **Synthesis of Programmatic Strategies**
- Research competition (Moraes 2021) to test ISS approaches for MicroRTS with a *fixed strategy model*
- MicroRTS strategy model = *CFG*

$$\begin{aligned} S_1 &\rightarrow C S_1 \mid S_2 S_1 \mid S_3 S_1 \mid \epsilon \\ S_2 &\rightarrow \text{if } (S_5) \text{ then } \{C\} \mid \text{if } (S_5) \text{ then } \{C\} \text{ else } \{C\} \\ S_3 &\rightarrow \text{for (each unit } u) \{S_4\} \\ S_4 &\rightarrow C S_4 \mid S_2 S_4 \mid \epsilon \\ S_5 &\rightarrow \text{not } B \mid B \\ B &\rightarrow b_1 \mid b_2 \mid \dots \mid b_m \\ C &\rightarrow c_1 C \mid c_2 C \mid \dots \mid c_n C \mid c_1 \mid c_2 \mid \dots \mid c_n \mid \epsilon \end{aligned}$$

Figure 16: The production rules of a context-free grammar (CFG) describing the strategy model for MicroRTS.

Performance Measure

- *win rate* (against fixed set of test scripts)

Interpretability Measure

- Inversely proportional to *number of statements*

Interpretability Measure

- Inversely proportional to *number of statements*
- *No justification* for use! → proposed study in RQ3b

Learning MicroRTS Strategies using ASP

RQ3(a)

How does an ASP-based approach towards developing a synthesizer for the *SynProS competition* compare to other synthesizers in this competition?

Answer Set Programming (ASP)

- Answer Set Programming ASP

Answer Set Programming (ASP)

- Answer Set Programming ASP
- ASP → *declarative programming* paradigm (like Prolog)

Answer Set Programming (ASP)

- Answer Set Programming ASP
- ASP → *declarative programming* paradigm (like Prolog)
- Can *model* and *generate* game levels (Smith and Mateas 2011; Smith, Andersen, et al. 2012)

Answer Set Programming (ASP)

- Answer Set Programming ASP
- ASP → *declarative programming* paradigm (like Prolog)
- Can *model* and *generate* game levels (Smith and Mateas 2011; Smith, Andersen, et al. 2012)
- Can model and generate *optimized* data viz. layouts (Moritz et al. 2018)

Learning MicroRTS Strategies using ASP

- MicroRTS strategy model (CFG) $\xrightarrow{\text{convert}}$ ASP model

Learning MicroRTS Strategies using ASP

- MicroRTS strategy model (CFG) $\xrightarrow{\text{convert}}$ ASP model
- MicroRTS strategy $\xrightarrow{\text{encode}}$ $\langle f_{\theta,1}, f_{\theta,2}, \dots, f_{\theta,i} \rangle$ using predicate vocabulary θ

Learning MicroRTS Strategies using ASP

- MicroRTS strategy model (CFG) $\xrightarrow{\text{convert}}$ ASP model
- MicroRTS strategy $\xrightarrow{\text{encode}}$ $\langle f_{\theta,1}, f_{\theta,2}, \dots, f_{\theta,i} \rangle$ using predicate vocabulary θ
- Train a *linear model* (\mathcal{L}) to *predict* win rate given feature encoding

Learning MicroRTS Strategies using ASP

- MicroRTS strategy model (CFG) $\xrightarrow{\text{convert}}$ ASP model
- MicroRTS strategy $\xrightarrow{\text{encode}}$ $\langle f_{\theta,1}, f_{\theta,2}, \dots, f_{\theta,i} \rangle$ using predicate vocabulary θ
- Train a *linear model* (\mathcal{L}) to *predict* win rate given feature encoding
- $\mathcal{L} \xrightarrow{\text{convert}}$ ASP constraints as in Moritz et al. (2018)

Learning MicroRTS Strategies using ASP

- MicroRTS strategy model (CFG) $\xrightarrow{\text{convert}}$ ASP model
- MicroRTS strategy $\xrightarrow{\text{encode}}$ $\langle f_{\theta,1}, f_{\theta,2}, \dots, f_{\theta,i} \rangle$ using predicate vocabulary θ
- Train a *linear model* (\mathcal{L}) to *predict* win rate given feature encoding
- $\mathcal{L} \xrightarrow{\text{convert}}$ ASP constraints as in Moritz et al. (2018)
- Evaluate resultant system using SynProS framework

Interpretability Factors for MicroRTS Strategies

- How to optimize MicroRTS strategy interpretability?

Interpretability Factors for MicroRTS Strategies

- How to optimize MicroRTS strategy interpretability?
- Which factors *most affect* MicroRTS strategy interpretability?

Interpretability Factors for MicroRTS Strategies

- How to optimize MicroRTS strategy interpretability?
- Which factors *most affect* MicroRTS strategy interpretability?

RQ3(b)

Which features of a MicroRTS strategy model have a statistically significant correlation with the interpretability of said strategy?

Task Design

- Conduct a *human-grounded* (Doshi-Velez and Kim 2017) evaluation

Task Design

- Conduct a *human-grounded* (Doshi-Velez and Kim 2017) evaluation
- Use a *forward simulation/prediction* task

Task Design

- Conduct a *human-grounded* (Doshi-Velez and Kim 2017) evaluation
- Use a *forward simulation/prediction* task
- Subjects presented with —

Task Design

- Conduct a *human-grounded* (Doshi-Velez and Kim 2017) evaluation
- Use a *forward simulation/prediction* task
- Subjects presented with —
 - Strategy

Task Design

- Conduct a *human-grounded* (Doshi-Velez and Kim 2017) evaluation
- Use a *forward simulation/prediction* task
- Subjects presented with —
 - Strategy
 - Game state (current)

Task Design

- Conduct a *human-grounded* (Doshi-Velez and Kim 2017) evaluation
- Use a *forward simulation/prediction* task
- Subjects presented with —
 - Strategy
 - Game state (current)
 - Options for future states (1 correct, 3 incorrect)

Task Design

- Conduct a *human-grounded* (Doshi-Velez and Kim 2017) evaluation
- Use a *forward simulation/prediction* task
- Subjects presented with —
 - Strategy
 - Game state (current)
 - Options for future states (1 correct, 3 incorrect)
- **Task:** predict expected future state from current state if strategy is followed and select option

Task Design

- Conduct a *human-grounded* (Doshi-Velez and Kim 2017) evaluation
- Use a *forward simulation/prediction* task
- Subjects presented with —
 - Strategy
 - Game state (current)
 - Options for future states (1 correct, 3 incorrect)
- **Task:** predict expected future state from current state if strategy is followed and select option
- Generate tasks using ASP model of MicroRTS strategy

Obtaining Significant Factors

Participant Experience			Strategy			Successful?	
Programming	RTS Games	...	$f_{\theta,1}$	$f_{\theta,2}$...	$f_{\theta,i}$	
:	:	:	:	:	:	:	:

Table 3: Sample dataset envisioned from study

- Train *decision tree* model to predict whether strategy will be correctly simulated

Obtaining Significant Factors

Participant Experience			Strategy			Successful?	
Programming	RTS Games	...	$f_{\theta,1}$	$f_{\theta,2}$...	$f_{\theta,i}$	
:	:	:	:	:	:	:	:

Table 3: Sample dataset envisioned from study

- Train *decision tree* model to predict whether strategy will be correctly simulated
- Obtain significant factors by measuring *Gini index* (Molnar 2018)

Proposed Work & Timeline

Semester	RQ	Task	Publication	Status
Spring '22	RQ1 RQ2(a)	ISS Framework Known Tactic Evaluation	(Krishnan and Martens 2022b)	Completed
Fall '22	RQ2(b)	Study	(Krishnan and Martens 2022a)	Completed
Spring '23	RQ2(c) RQ3(a) RQ3(a,b)	Study (<i>under review</i>) Dataset assembly MicroRTS ASP Model	(Krishnan, Martens, and Jhala 2023)	Completed In progress
Summer '23	RQ3(a) RQ3(b)	Study Task Design (Strategies)	Advances in Computer Games '23	
Fall '23	RQ3(b) RQ3(b)	Task Design IRB Approval		
Spring '24	RQ3(b) – –	Analysis Dissertation writing Thesis defence	World Conference on Explainable Artificial Intelligence '24	
Summer '24	–	Graduation		

Conclusion

- **Goal:** investigate approaches to the problem of *interpretable strategy synthesis* (ISS) for games

Conclusion

- **Goal:** investigate approaches to the problem of *interpretable strategy synthesis* (ISS) for games
- Defined a *framework* for ISS

Conclusion

- **Goal:** investigate approaches to the problem of *interpretable strategy synthesis* (ISS) for games
- Defined a *framework* for ISS
- Approached ISS using ILP for *chess*

Conclusion

- **Goal:** investigate approaches to the problem of *interpretable strategy synthesis* (ISS) for games
- Defined a *framework* for ISS
- Approached ISS using ILP for *chess*
- Proposal to approach ISS for *MicroRTS* using ASP

Conclusion

- **Goal:** investigate approaches to the problem of *interpretable strategy synthesis* (ISS) for games
- Defined a *framework* for ISS
- Approached ISS using ILP for *chess*
- Proposal to approach ISS for *MicroRTS* using ASP
- Expected outcomes –

Conclusion

- **Goal:** investigate approaches to the problem of *interpretable strategy synthesis* (ISS) for games
- Defined a *framework* for ISS
- Approached ISS using ILP for *chess*
- Proposal to approach ISS for *MicroRTS* using ASP
- Expected outcomes –
 - Benefit esports industry → better analytics for player performance

Conclusion

- **Goal:** investigate approaches to the problem of *interpretable strategy synthesis* (ISS) for games
- Defined a *framework* for ISS
- Approached ISS using ILP for *chess*
- Proposal to approach ISS for *MicroRTS* using ASP
- Expected outcomes –
 - Benefit esports industry → better analytics for player performance
 - Benefit explainable AI research → new ways to generate policy explanations

Thank You!

Questions?

References I

- Berliner, Hans J (1975). *A representation and some mechanisms for a problem solving chess program*. Tech. rep. Carnegie-Mellon Univ Pittsburgh PA Dept of Computer Science.
- Bramer, Max Arthur (1977). “Representation of Knowledge for Chess Endgames Towards a Self-Improving System”. PhD thesis. Open University (United Kingdom).
- Bratko, Ivan (1982). “Knowledge-based problem-solving in AL3”. In: *Machine intelligence* 10, pp. 73–100.

References II

Butler, Eric, Emina Torlak, and Zoran Popović (2017). “Synthesizing Interpretable Strategies for Solving Puzzle Games”. In: *Proceedings of the 12th International Conference on the Foundations of Digital Games*. FDG ’17. Hyannis, Massachusetts: Association for Computing Machinery. ISBN: 9781450353199. DOI: 10.1145/3102071.3102084. URL: <https://doi.org/10.1145/3102071.3102084>.

References III

- Canaan, Rodrigo et al. (2018). “Evolving Agents for the Hanabi 2018 CIG Competition”. In: *2018 IEEE Conference on Computational Intelligence and Games (CIG)*, pp. 1–8. DOI: 10.1109/CIG.2018.8490449.
- Doshi-Velez, Finale and Been Kim (2017). *Towards A Rigorous Science of Interpretable Machine Learning*. arXiv: 1702.08608 [stat.ML].

References IV

- Freitas, João Marcos de, Felipe Rafael de Souza, and Heder S. Bernardino (2018). “Evolving Controllers for Mario AI Using Grammar-based Genetic Programming”. In: *2018 IEEE Congress on Evolutionary Computation (CEC)*, pp. 1–8. DOI: 10.1109/CEC.2018.8477698.
- Gebser, Martin et al. (2015). “Abstract gringo”. In: *Theory and Practice of Logic Programming* 15.4-5, pp. 449–463.
- Gelfond, Michael and Vladimir Lifschitz (1988). “The stable model semantics for logic programming.”. In: *ICLP/SLP*. Vol. 88. Cambridge, MA, pp. 1070–1080.

References V

Gobet, Fernand and Peter J Jansen (2006). “Training in chess: A scientific approach”. In: *Education and chess*.

Huberman, Barbara Jane (Aug. 1968). “A program to play chess end games”. PhD thesis. Department of Computer Science, Stanford University.

References VI

Krishnan, Abhijeet and Chris Martens (Oct. 2022a). “Synthesizing interpretable chess tactics from player games”. In: *Proceedings of the Workshop on Artificial Intelligence for Strategy Games (SG) and Esports Analytics (EA), 18th AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment*. American Association for Artificial Intelligence.

References VII

- Krishnan, Abhijeet and Chris Martens (Mar. 2022b). “Towards the automatic synthesis of interpretable chess tactics”. In: *Proceedings of the Explainable Agency in Artificial Intelligence Workshop, 36th AAAI Conference on Artificial Intelligence*. American Association of Artificial Intelligence, pp. 91–97.
- Krishnan, Abhijeet, Chris Martens, and Arnav Jhala (Mar. 2023). “Improving strategy synthesis for chess using precision and recall”. In: [Manuscript submitted for publication].

References VIII

- Mariño, Julian R. H. et al. (May 2021). “Programmatic Strategies for Real-Time Strategy Games”. In: *Proceedings of the AAAI Conference on Artificial Intelligence* 35.1, pp. 381–389. DOI: 10.1609/aaai.v35i1.16114. URL: <https://ojs.aaai.org/index.php/AAAI/article/view/16114>.
- Mariño, Julian RH and Claudio FM Toledo (2022). “Evolving interpretable strategies for zero-sum games”. In: *Applied Soft Computing* 122, p. 108860.

References IX

- Medeiros, Leandro C, David S Aleixo, and Levi HS Lelis (2022). “What can we Learn Even From the Weakest? Learning Sketches for Programmatic Strategies”. In: *arXiv preprint arXiv:2203.11912*.
- Mesentier Silva, Fernando de et al. (2016). “Generating heuristics for novice players”. In: *2016 IEEE Conference on Computational Intelligence and Games (CIG)*. IEEE, pp. 1–8.
- Molnar, Christoph (2018). “A guide for making black box models explainable”. In:
<https://christophm.github.io/interpretable-ml-book>, p. 3.

References X

- Moraes, Rubens (July 2021). *SynProS - Synthesis of Programmatic Strategies*. URL:
<https://rubensolv.github.io/synpros-microrts/> (visited on 03/26/2023).
- Morales, Eduardo (1992). “First order induction of patterns in Chess”. PhD thesis. PhD thesis, The Turing Institute-University of Strathclyde.
- Moritz, Dominik et al. (2018). “Formalizing visualization design knowledge as constraints: Actionable and extensible models in draco”. In: *IEEE transactions on visualization and computer graphics* 25.1, pp. 438–448.

References XI

- Ontanon, Santiago (June 2021). “The Combinatorial Multi-Armed Bandit Problem and Its Application to Real-Time Strategy Games”. In: *Proceedings of the AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment* 9.1, pp. 58–64. DOI: 10.1609/aiide.v9i1.12681. URL: <https://ojs.aaai.org/index.php/AIIDE/article/view/12681>.
- Pitrat, Jacques (1977). “A chess combination program which uses plans”. In: *Artificial Intelligence* 8.3, pp. 275–321.

References XII

- Smith, Adam M, Erik Andersen, et al. (2012). “A case study of expressively constrainable level design automation tools for a puzzle game”. In: *Proceedings of the International Conference on the Foundations of Digital Games*, pp. 156–163.
- Smith, Adam M and Michael Mateas (2011). “Answer set programming for procedural content generation: A design space approach”. In: *IEEE Transactions on Computational Intelligence and AI in Games* 3.3, pp. 187–200.

References XIII

- Spronck, Pieter, Ida Sprinkhuizen-Kuyper, and Eric Postma (2004). “Online adaptation of game opponent AI with dynamic scripting”. In: *International Journal of Intelligent Games and Simulation* 3.1, pp. 45–53.
- Szabo, Alexander (1984). “Computer chess tactics and strategy”. PhD thesis. University of British Columbia. DOI: <http://dx.doi.org/10.14288/1.0051870>. URL: <https://open.library.ubc.ca/collections/ubctheses/831/items/1.0051870>.
- Wilkins, David Edward (1979). *Using patterns and plans to solve problems and control search*. Stanford University.

References XIV

Zhang, Yu et al. (Oct. 2021). “A Survey on Neural Network Interpretability”. In: *IEEE Transactions on Emerging Topics in Computational Intelligence* 5.5, pp. 726–742. ISSN: 2471-285X. DOI: 10.1109/TETCI.2021.3100641.

Strategy (σ)

Definition (Strategy)

Given a game environment \mathcal{G} modeled as a finite, episodic MDP $\langle \mathcal{S}, \mathcal{A}, \mathcal{P}, \mathcal{R}, \gamma \rangle$, a *strategy* σ is —

$$\sigma(a|s) \doteq \mathbb{P}[A_t = a | S_t = s], \forall s \in A_\sigma, a \in \mathcal{A}(s)$$

A_σ : set of *applicable* states

◀ Return

Divergence

Move Evaluation Function

Given chess engine E with position evaluation function $v_E(s)$, we can obtain a move evaluation function $q_E(s, a)$ as —

$$q_E(s, a) = \sum_{s', r} \mathcal{P}(s', r | s, a) [r + v_E(s')] \quad (1)$$

$$= v_E(s'), s' \text{ is non-terminal} \quad (2)$$

Equation 2 follows from 1 since rewards in chess are 0 for non-terminal states, $\gamma = 1$, and chess rules are deterministic.

Divergence

Difference Function

Given two moves a_1, a_2 made in a position s , we can calculate their difference $d_E(s, a_1, a_2)$ as —

$$d_E(s, a_1, a_2) \doteq | q_E(s, a_1) - q_E(s, a_2) | \quad (3)$$

◀ Return

Divergence

Definition (Divergence)

Divergence of a tactic from a set of examples P is the average difference in *evaluation* between the moves suggested by the tactic and the ground truth move.

$$\text{Divergence}_E(\sigma, P) \doteq \frac{1}{|P_A|} \sum_{(s, a_1) \in P_A} \sum_{a_2 \in \mathcal{A}(s)} \sigma(a_2 | s) d_E(s, a_1, a_2) \quad (4)$$

◀ Return

PAL

- Patterns and Learning (Morales 1992)
- ILP system to learn chess *patterns*
- Predicate vocabulary
- *rIgg* algorithm + heuristics to learn patterns
- Automatic *example generator* to learn target concepts

◀ Return

Inductive Logic Programming

- *symbolic* machine learning technique
- ILP problem $\langle E^+, E^-, B \rangle$
 - E^+ : positive examples (of concept)
 - E^- : negative examples (of concept)
 - B : background knowledge
- **Goal:** *induce* hypothesis that entails (fits) E^+ but not E^-

◀ Return

Target Concept

$$\begin{aligned}
 E^+ &= \left\{ \begin{array}{l} \text{last}([m, a, c, h, i, n, e], e). \\ \text{last}([l, e, a, r, n, i, n, g], g). \\ \text{last}([a, l, g, o, r, i, t, h, m], m). \end{array} \right\} \\
 E^- &= \left\{ \begin{array}{l} \text{last}([m, a, c, h, i, n, e], m). \\ \text{last}([m, a, c, h, i, n, e], c). \\ \text{last}([l, e, a, r, n, i, n, g], x). \\ \text{last}([l, e, a, r, n, i, n, g], i). \end{array} \right\} \\
 B &= \left\{ \begin{array}{l} \text{empty}(A) :- \dots \\ \text{head}(A, B) :- \dots \\ \text{tail}(A, B) :- \dots \end{array} \right\}
 \end{aligned}$$

Possible Hypothesis

$$H = \left\{ \begin{array}{l} \text{last}(A, B) :- \text{head}(A, B), \text{tail}(A, C), \text{empty}(C). \\ \text{last}(A, B) :- \text{tail}(A, C), \text{last}(C, B). \end{array} \right\}$$
[◀ Return](#)

Precision/Recall-based Constraints

Definition (Precision constraint)

A precision constraint prunes the specializations of a hypothesis if its precision on a set of examples is less than some pre-defined lower limit.

Definition (Recall constraint)

A recall constraint prunes specializations of a hypothesis if its recall on a set of examples is less than some pre-defined lower limit.

◀ Return

Precision/Recall-based Constraints

Theorem

Given hypotheses $H_1, H_2 \in \mathbb{H}$ with $H_1 \preceq H_2$ and having recall values of r_1 and r_2 on a training set respectively, then $r_1 \leq r_2$.

◀ Return

Predicate Vocabulary

- Allows more *situational rule* expression – en passant, promotion
- Allows *more efficient* unification

◀ Return

Answer Set Programming

- *Declarative programming* paradigm based on *stable models* (Gelfond and Lifschitz 1988)
- ASP language (Gebser et al. 2015) allows using rules to —
 - *model* a design space
 - *restrict* it using integrity constraints
 - *generate* instances in the newly restricted space

◀ Return

Example

```
#const width=10.

param("width",width).

dim(1..width).

tile((X,Y)) :- dim(X), dim(Y).

adj((X1,Y1),(X2,Y2)) :- tile((X1,Y1)), tile((X2,Y2)), \\
#abs(X1-X2)+#abs(Y1-Y2) == 1.

start((1,1)). finish((width,width)).

% tiles have at most one named sprite
0 { sprite(T,wall;gem;altar) } 1 :- tile(T).

% there is exactly one altar and one gem in the whole level
:- not 1 { sprite(T,altar) } 1. :- not 1 { sprite(T,gem) } 1.
```

Figure 17: An ASP program which can generate maze-like levels with integrity constraints that specify the number of game objects.