



Search or jump to...

Pull requests Issues Marketplace Explore



GAIGResearch / TabletopGames

Unwatch 4

Unstar 3

Fork 0

Code Issues 9 Pull requests Actions Projects 3 Wiki Security Insights

master 4 branches 1 tag

Go to file

Add file

Code

rdgain	Game images	0de1054	11 seconds ago	398 commits
data	Game images		11 seconds ago	
src/main/java	Hotfixes		19 hours ago	
.gitignore	Printing games in the framework and their properties		4 months ago	
README.md	Updated Readme.md, image is missing for Pandemic		13 days ago	
pom.xml	AI & Game initial measurements, some plotting (#106)		3 months ago	

About

No description, website, or topics provided.

Readme

Releases 1

v1.0 Latest on Aug 25

< TAG />

# A Tabletop Games Framework

Raluca D. Gaina, Martin Balla, Alexander Dockhorn, Raul Montoliu, Diego Perez-Liebana

EXAG  
2020



# Modern Tabletop Games

## What?

- Board games
- Card games
- Dice games
- Role-playing games
- Pen and paper games
- ...



## Why?

- High complexity: many components / rules / players / types of interaction
- Partial observability: facedown decks / player hands of cards / secret objectives
- Diverse/asymmetric player roles
- Control parameters: how many cards per player / how many actions in a turn / value of coins
- Unique game state representations
- Cooperation combined with competition
- Large action spaces

# A look to the past (and present?)

---

## Traditional board games

- Chess, Othello, Go etc.

## Tabletop games

- Card games: Poker (Moravcik et al. 2017), Bridge (Cazenave and Ventos 2019), Hannabi (Bard et al. 2020)
- Asymmetric player roles: The Resistance (Serrino et al. 2019), Ultimate Werewolf (Eger and Martens 2019)
- Strategic complexity: Pandemic (Chacon and Eger 2019; Sfikas and Liapis 2020)
- Large action spaces: Bloodbowl (Justesen et al. 2019)
- Statistical forward planning applications:
  - MCTS in Settlers of Catan (Szita, Chaslot and Spronck 2009), Risk (Gibson, Desai, and Zhao 2010)
  - RHEA in Splendor (Bravi et al. 2019)
- Play-testing in Ticket to Ride (de Mesentier Silva et al. 2017)
- Procedural Content Generation in TTRPGs (Guzdial et al. 2020)

# A look to the past (and present?)

## Description-language-based frameworks

- General Game Playing (GGP) (Genesereth, Love, and Pell 2005)
- Ludii (Piette et al. 2019)
- Regular boardgames (RBG) (Kowalski et al. 2019)

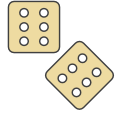
## Freeform frameworks

- Tabletop Simulator (Henry 2015)
- OpenSpiel (Lanctot et al. 2019)



# TAG – 3 motivating pillars

---



## Modern tabletop games as complex AI challenges

- The “Why” before



## General game playing of modern tabletop games

- Common API for games and AI players



## Facilitating community-driven database for AI research

- Features supporting easy development of new games and AI players under the same common platform

# TAG – the technical bit

---

## Java

- No, it's not Python (yet?).
- Yes, it runs fast.
- No game description language, all coded.

## Features of interest at a glance

- *Abstract classes* for game/AI skeletons
- Add-on *interfaces* for automatic optimisation of parameters, custom observations
- Ready-made common *rules* / *actions* / *components* (+ extendable)
- *Prototyping GUI* for immediate running and interacting with the game, mid-development
  - ... and an easily extendable template for custom displays and interactions
- Game *tagging* / categorisation
- *8 games* implemented, more in development
- *General AI players* compatible with all games (various performance...)
- Fully functioning *game loop*, *game analysis*

# TAG – tabletop game concepts

## Concepts

- **Action:** things players do
- **Rule:** things the game does
- **Turn order:** defines order of players
- **Game phase:** time frames with specific rules/actions
- **Components:** game objects/pieces, what actions are rules modify

## Components

- Tokens
- Cards
- Dice
- Counters
- Grid boards
- Graph boards
- **Decks:** ordered lists of components
- **Areas:** mapping from component ID to component



# TAG – core classes

---

## Game state (GS)

- Container class, made up of game components + variables
- Describes a moment in time
- Defines component access methods and scoring functions (optional)
- Can be *copied*
- A *reduced copy* is available to AI players as an observation (with hidden information in partial observable modes)

## Forward model (FM)

- Logic class, controls the rules which modify a given game state
- Sets up the initial state of the game
- Decides which actions are available in a given game state
- Applies player actions and game rules to advance the game state
- Checks any end of game conditions
- Is available to AI players for game *simulations*



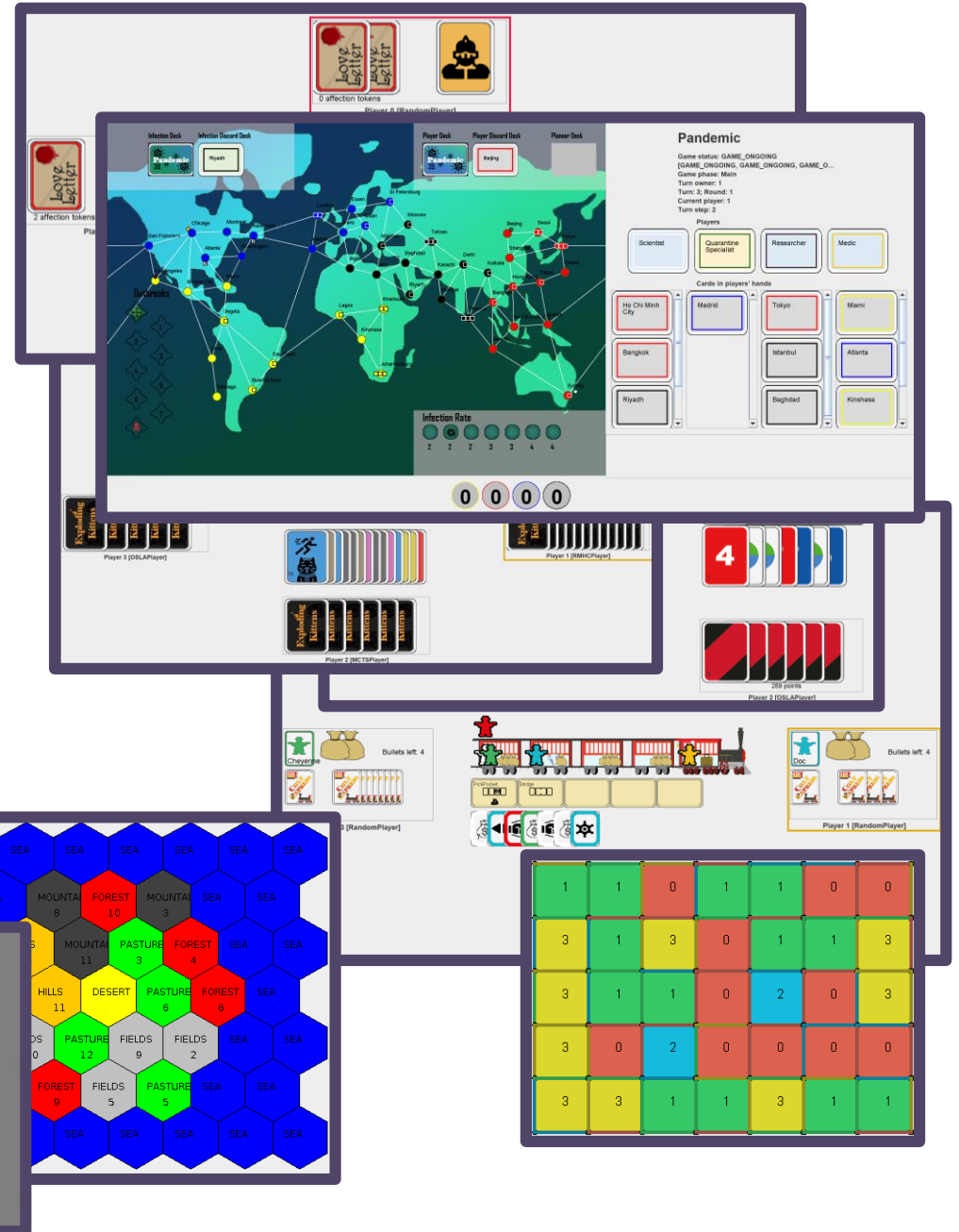
# TAG – games

## Games implemented

- Tic-Tac-Toe
- Dots & Boxes
- Love Letter (Kanai 2012)
- Uno (Robbins 1971)
- Virus! (Cabrero and others 2015)
- Exploding Kittens (Inman and others 2015)
- Colt Express (Raimbault 2014)
- Pandemic (Leacock 2008)

## Games in progress

- Descent (Fantasy Flight Publishing, Inc. 2012)
- Carcassonne (Wrede 2000)
- Settlers of Catan (Teuber 1995)



# TAG – players

---

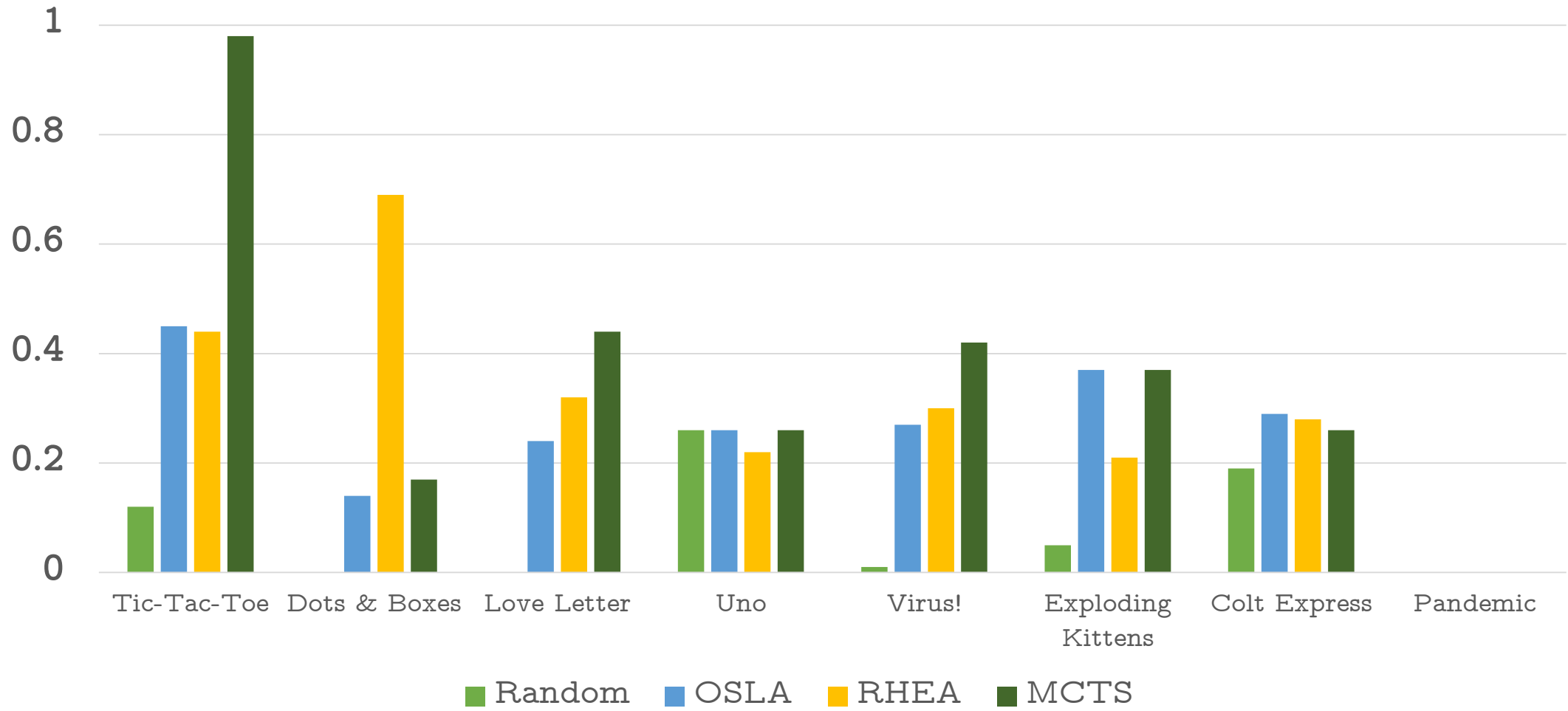
## Human play

- **Console:** use keyboard to enter console input and read printed game states
- **GUI:** use action buttons (or more complex custom interactions designed per game) to directly interact with the game and observe a visual representation of the game state

## Automatic players

- **Random:** randomly chooses one of the available actions
  - **One Step Look Ahead (OSLA):** exhaustively tries all possible actions, simulating their effect with the FM, and chooses the one which leads to the game state with the highest *score*
  - **Rolling Horizon Evolutionary Algorithm (RHEA):** evolves a sequence of actions, using the FM to simulate their effect and chooses the first action of the sequence which led to the highest *score*
  - **Monte Carlo Tree Search (MCTS):** builds an asymmetric game tree, using the FM to simulate the effect of actions and build statistics of observed *scores*; chooses the most visited child from the root
- \* *Game state evaluation:* uses scoring functions defined in the games, but can be swapped for other heuristic functions.

# AI Player Performance



# A look to the future

---

- Competitive, cooperative, mixed games
- Hidden information, belief systems
- Dynamic/changing rules
- Asymmetric player roles
- Role-playing, strategy, campaign games
- Parameter optimisation
- Observation diversity: object-based, vector, feature-based



# Takeaways

---



## Tabletop games

- Exciting opportunities for research



## TAG: Java framework for tabletop games

- Providing common API for implementing both games and AI players



## Open-source framework

- <https://github.com/GAIGResearch/TabletopGames>

# < TAG />

## A Tabletop Games Framework

Raluca D. Gaina, Martin Balla, Alexander Dockhorn, Raul Montoliu, Diego Perez-Liebana

Thank you for watching!

### More about TAG




<https://tinyurl.com/TAG-EXAG>

### More about GAIG @ QMUL





<http://gameai.eecs.qmul.ac.uk>

 @GameAI\_QMUL

### More about me



<http://rdgain.github.io>

 @b\_gum22  
 r.d.gaina@qmul.ac.uk

