Game Design with Answer Set Programming

DaBlocksWorld

Dario Klepoch

17.03.2020

Universität Potsdam

AI-based Game Design

"the development of innovative artificial intelligence systems plays a crucial role in the exploration of currently unreachable [game design] spaces."

Elad hari et al. (2011)

Contents

Godot

Godot

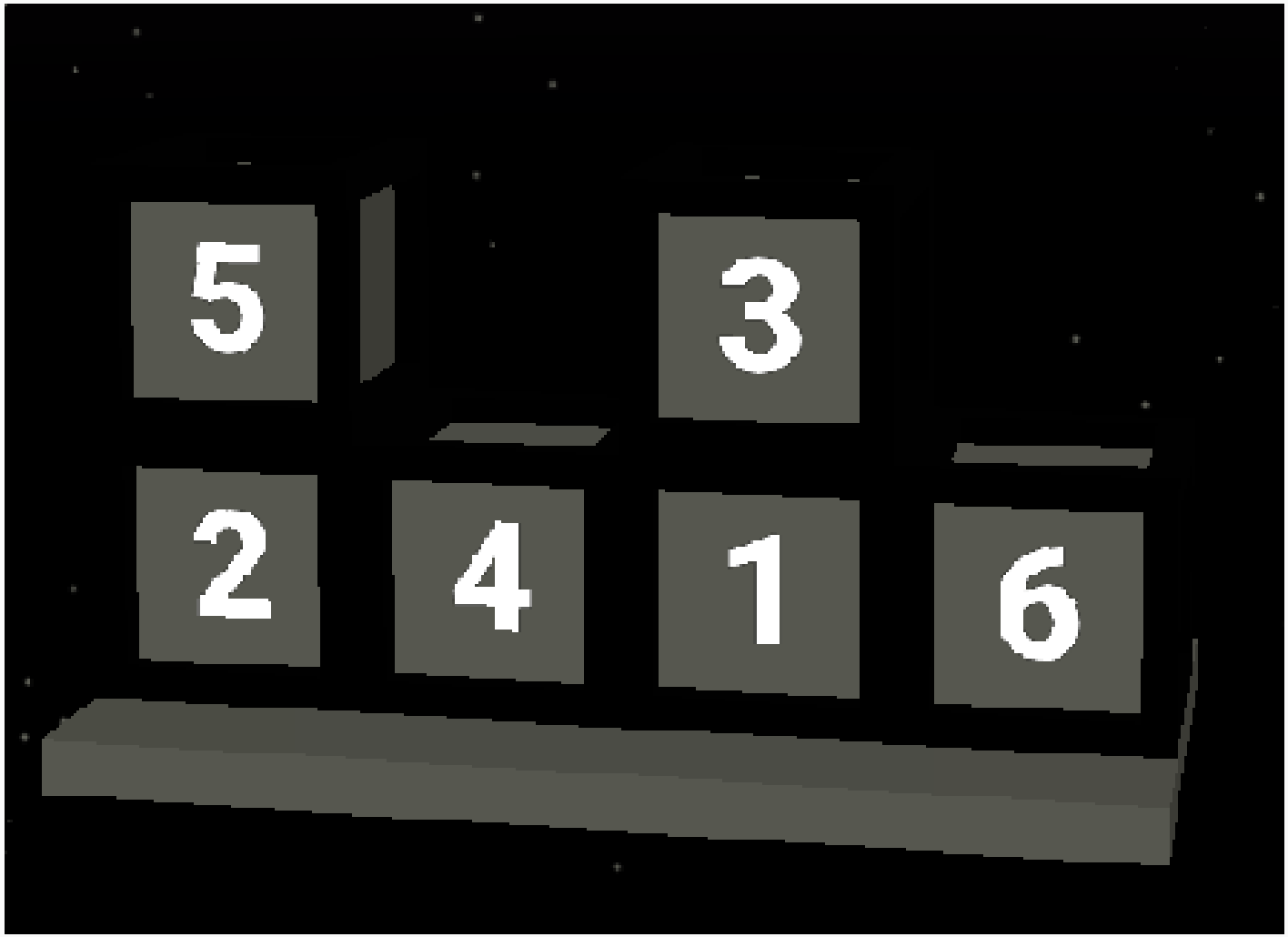
- Open source game engine
- Provides tool set for 2D/3D development
- Under MIT license
- Written in C/C++
- Used for this project
- Supports C/C++/C#/GodotScript

Blocks World Planning Problem

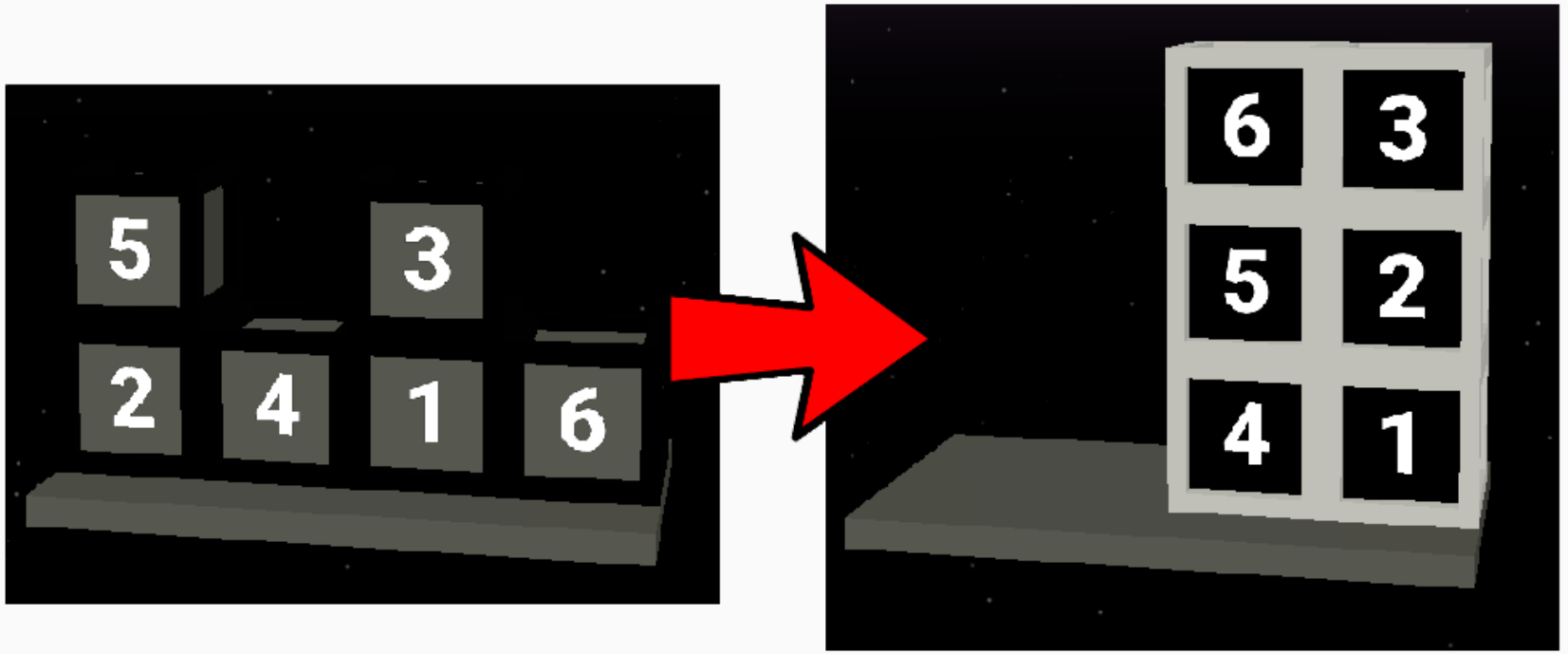
BWPP

- Blocks World Planning Problem (BWPP)
- Consists of Blocks
- Can be placed on the ground
- Or on another Block (creates Stack)
- Has start-configuration and goal-configuration
- Goal:
    - Convert start-config to goal-config (with few moves)
- Ambiguous BWPP configurations exist

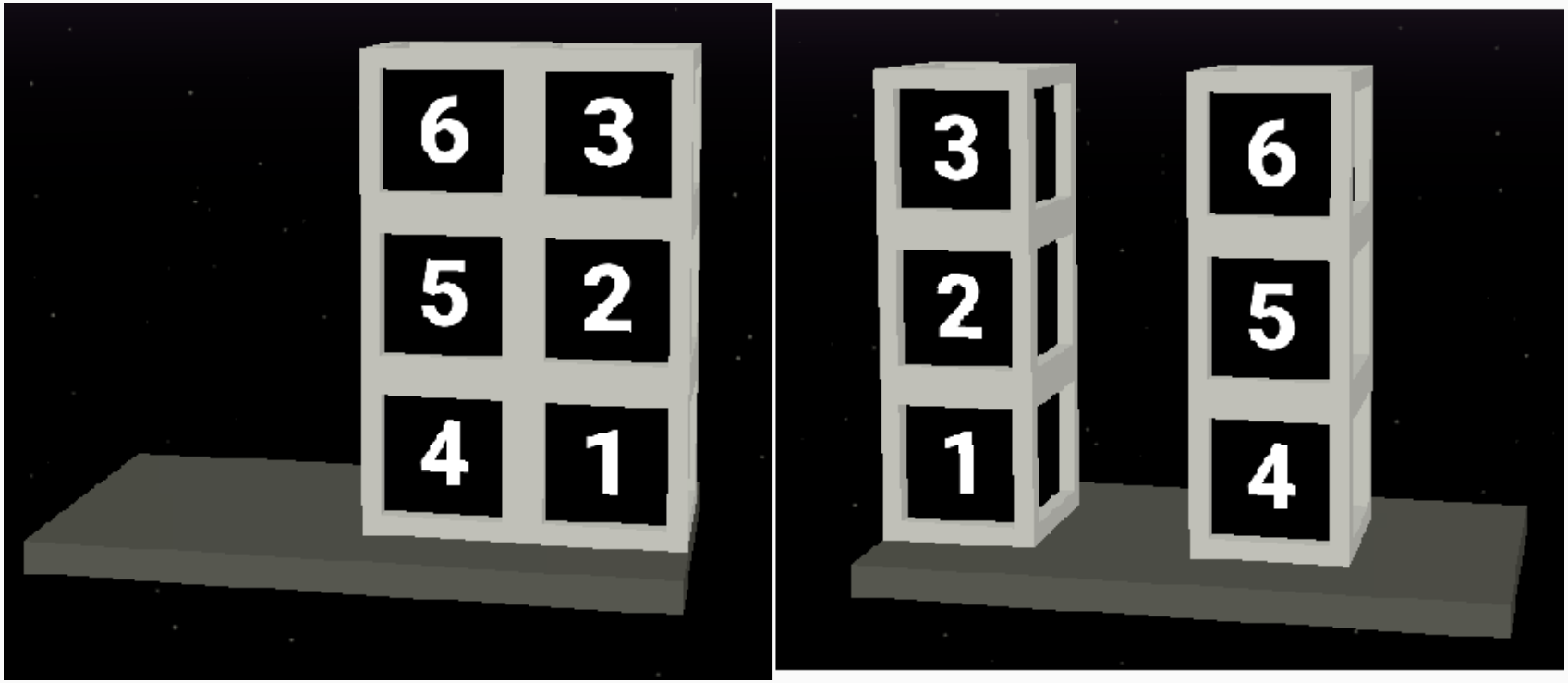
Start-config



Start-/goal-config



Ambiguous goal-config



Rules for moves

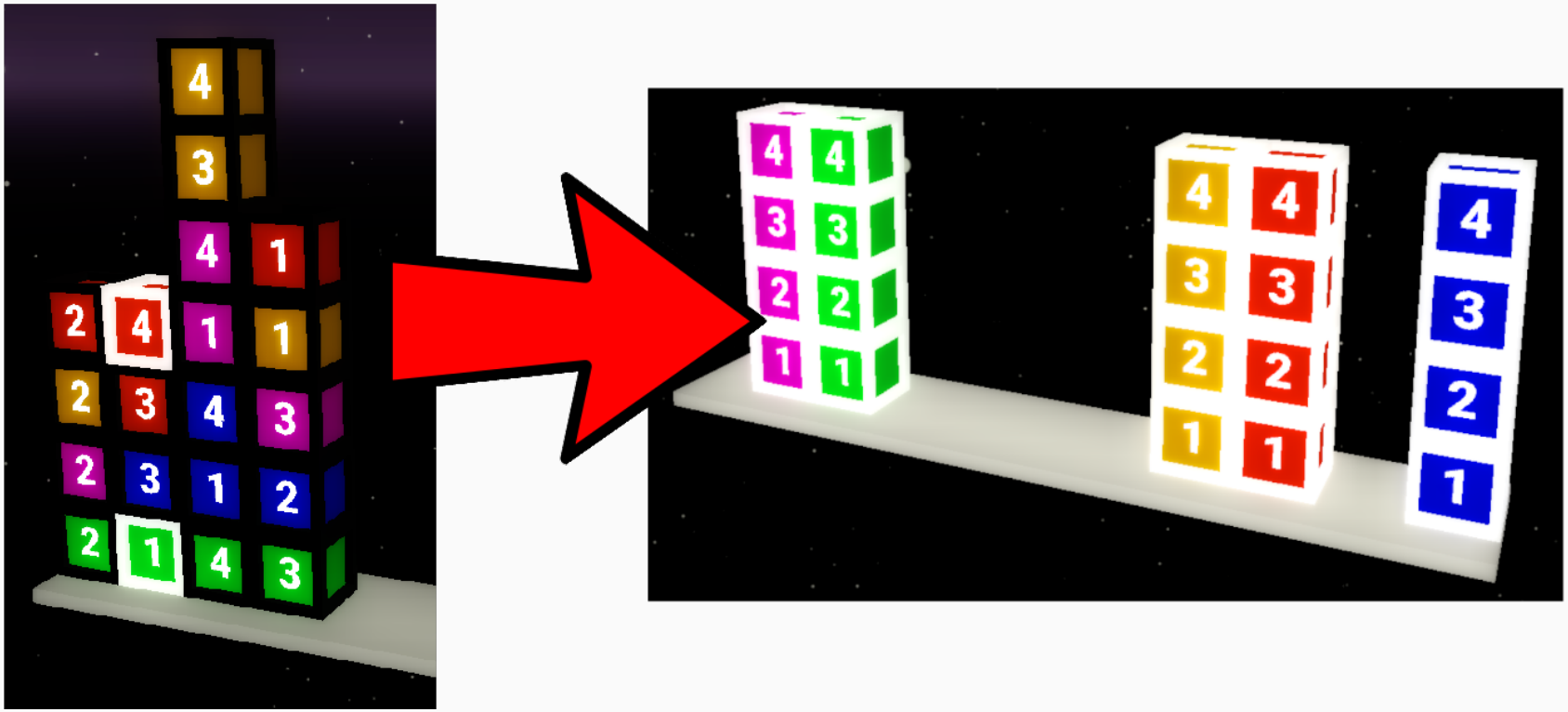
- Legal moves follow these rules:
  - No other block on the moved block
  - Only one block can be moved at one time point
  - Block can be moved to the ground (without limit)

Game Design

What is DaBlocksWorld?

- Is an interactive implementation of BWPP
- The differences are:
  - Blocks cannot be moved unlimited to the ground
  - Blocks cannot be stacked unlimited
  - Width and height match optimal plan
  - Numbers are limited
  - Stacks are marked by colors
- Developed in Godot
- Uses Clingo for inference tasks

Changed rules



Game Loop of DaBlocksWorld

- Player selects difficulty
- Random configuration will be generated
- Translate configuration to Clingo atoms
- Solve with Clingo
- Let player beat the level
- Rate player based on the number of optimal moves

Show game cycle

Solving BWPP

Solving Configurations

```
#include <incmode>.
#program base.
% DOMAIN
do(X,Z)   :- init(X,Y), not table(Y), table(Z).
do(X,Y)   :- goal(X,Y), not table(Y).
on(X,Y,0) :- init(X,Y).
#program check(t).
% TEST
:- query(t), goal(X,Y),  not on(X,Y,t).
#program step(t).
% GENERATE
1 {move(X,Y,t) : do(X,Y)} 1.
% DEFINE
move(X,t)   :- move(X,Y,t).
on(X, Y, t) :- move(X,Y,t).
on(X, Y, t) :- on(X,Y,t-1), not move(X,t).
...
```

Relevant moves

```
#program base.
% DOMAIN
do(X,Z)    :- init(X,Y), not table(Y), table(Z).
do(X,Y)    :- goal(X,Y), not table(Y).
on(X,Y,0)  :- init(X,Y)
```

Generating moves

```
#program step(t).
% GENERATE
1 {move(X,Y,t) : do(X,Y)} 1.
```

Needed height and width

```
blocksOnGround(A, t) :- #count{X :
                                on(X, 0, t)} = A.
height(B, 1, t) :- on(B, 0, t).
height(B1, H+1, t) :- height(B2, H, t),
                      on(B1, B2, t),
                      amountOfBlocks(A), H < A.
```

The goal condition

```
#program check(t).
% TEST
:- query(t), goal(X,Y), not on(X,Y,t).
```

Clingo's output

```
% DISPLAY
#show move/3.
#show blocksOnGround/2.
#show height/3.
```

Conclusion

Conclusion

- ASP (AI) enables new game design aspects
- We could rate the performance of the player
- An algorithm without ASP would be larger and most probably slower

Questions

Questions?

Sources

- https://godotengine.org/
- https://potassco.org/clingo/
- https://github.com/CaptainDario/DaBlocksWorld
- https://aaai.org/ojs/index.php/aimagazine/article/view/2673
- https://pngio.com/images/png-a891433.html

Thank you

Thank you for your attention!