School of Computer Science & Software Engineering

HYPANT
A Hypergame Analysis Tool

Lachlan Brumley
Bachelor of Software Engineering

Supervisor: Dr Carlo Kopp

Monash University, Clayton, Australia
email: lnbru1@student.monash.edu.au

Project Goal

To create a software tool that could

- Perform the analysis of hypergames.
- Allow users to create and store hypergames.

Game Theory

- Developed by von Neumann and Morgenstern in 1950s
- Used to mathematically study games of strategy
- Defined a game as:
  - *Players*
  - *Rules that define the players' actions*
  - *Moves that a player can make*
- Players are considered to be rational
- Each game has one or more equilibriums

Game Analysis Algorithm

- Created by Fraser & Hipel in the 1980s
- Works on an extended game model
- Compares each player's preferences to calculate which outcomes are stable for each player.
- Outcomes that are stable for all players are possible equilibriums for that game.

Expanded Game Definition

A Game consists of:

- Players
  The groups or individuals involved in the conflict.
- Options
  The various actions that players may take. Options are either taken or not taken.
- Outcomes
  An outcome is a combination of all of the options, with each option either chosen or not chosen.
- Preferences
  Each player ranks the outcomes in their preferred order.

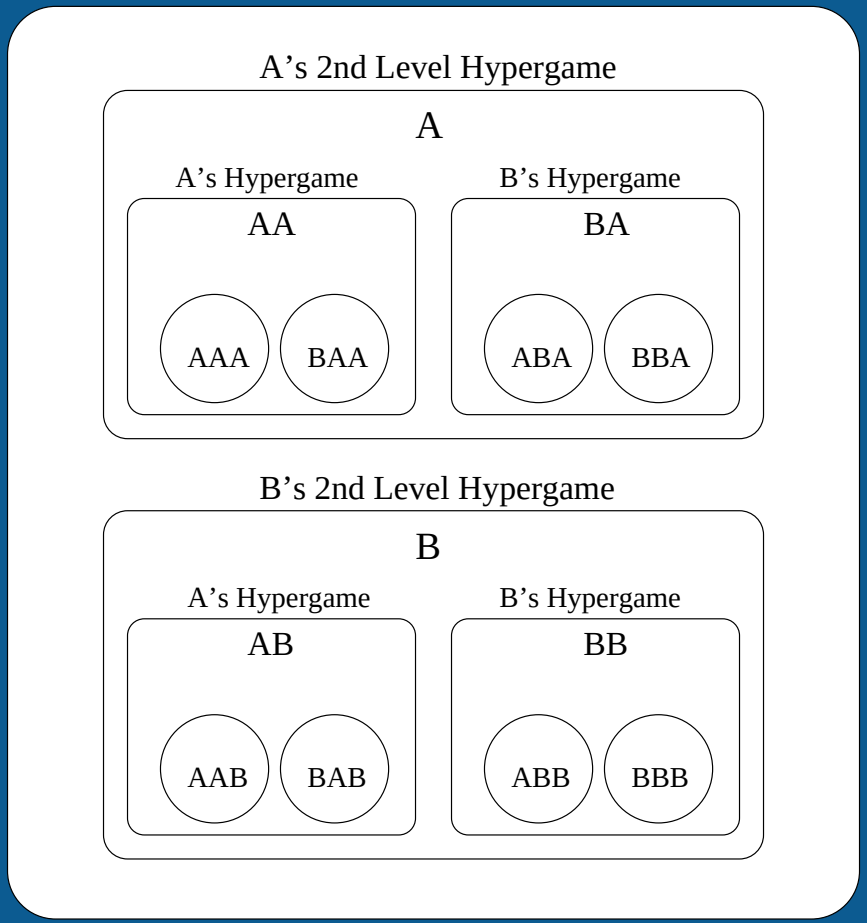
Shortcomings of Games

- Games depend on perfect perception.
- Can only model situations where each player has a perfect understanding of the situation, which doesn't always occur.
- Many disasters in the past have been caused by people having incomplete or incorrect information about their situation.

Hypergames

- Uses a number of games to model different perceptions of a situation.
- By using different games to model the conflict, conflicts where players do not have perfect percerptions can be modeled.
- Hypergames allow the modeling of strategic surprise and deception by players.
- Hypergames can model situations where players
  - Do not correctly perceive their opponents
  - Do not correctly perceive their options or those of their opponent's
  - Incorrectly perceive the preferences of their opponent.

3rd Level Hypergame



HYPANT

- Hypergame Analysis Tool
- Uses Fraser and Hipel's game analysis algorithm
- Capable of
  - Reading and writing hypergames to/from files.
  - Displaying hypergames to users.
  - Analysing individual games.
  - Analysing hypergames.
  - Writing output of analyses to screen and file.

Hypergame Modeling Language

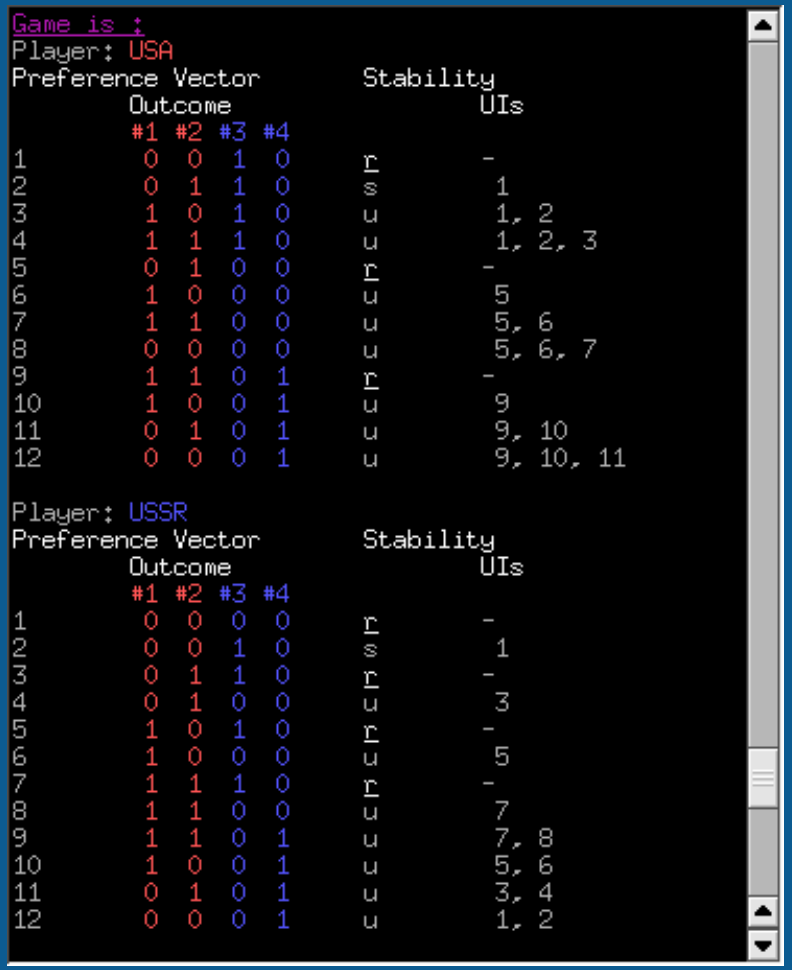
- Need a method to describe hypergames for loading and saving them.
- A custom language allows for the use of human readable input files.
- Language can be reused in future programs.
- Easier to read in and error check than other methods.
- Games are parsed and the information put into custom datastructures, ready for analysis.
- Parser implemented with flex and bison.

HML Example

```
%%Cuban Missile Crisis Game
{
 {
  {}, %% Game Perception
  {
   { %% Player Name & Options
    USA,{{#1,Air Strike},{#2,Blockade}},{},
    { %%Preference Vector
     {#1=0,#2=0,#3=1,#4=0},{#1=0,#2=1,#3=1,#4=0},{#1=1,#2=0,#3=1,#4=0},
     {#1=1,#2=1,#3=1,#4=0},{#1=0,#2=1,#3=0,#4=0},{#1=1,#2=0,#3=0,#4=0},
     {#1=1,#2=1,#3=0,#4=0},{#1=0,#2=0,#3=0,#4=0},{#1=1,#2=1,#3=0,#4=1},
     {#1=1,#2=0,#3=0,#4=1},{#1=0,#2=1,#3=0,#4=1},{#1=0,#2=0,#3=0,#4=1}
    }
   },
   { %% Player Name & Options
    USSR,{{#3,Withdraw},{#4,Escalate}},{{#3,#4}},
    { %%Preference Vector
     {#1=0,#2=0,#3=0,#4=0},{#1=0,#2=0,#3=1,#4=0},{#1=0,#2=1,#3=1,#4=0},
     {#1=0,#2=1,#3=0,#4=0},{#1=1,#2=0,#3=1,#4=0},{#1=1,#2=0,#3=0,#4=0},
     {#1=1,#2=1,#3=1,#4=0},{#1=1,#2=1,#3=0,#4=0},{#1=1,#2=1,#3=0,#4=1},
     {#1=1,#2=0,#3=0,#4=1},{#1=0,#2=1,#3=0,#4=1},{#1=0,#2=0,#3=0,#4=1}
    }
   }
  }
 }
}
```

Screenshots

Figure 1: Equilibriums for Cuban Missile Crisis



Game is :

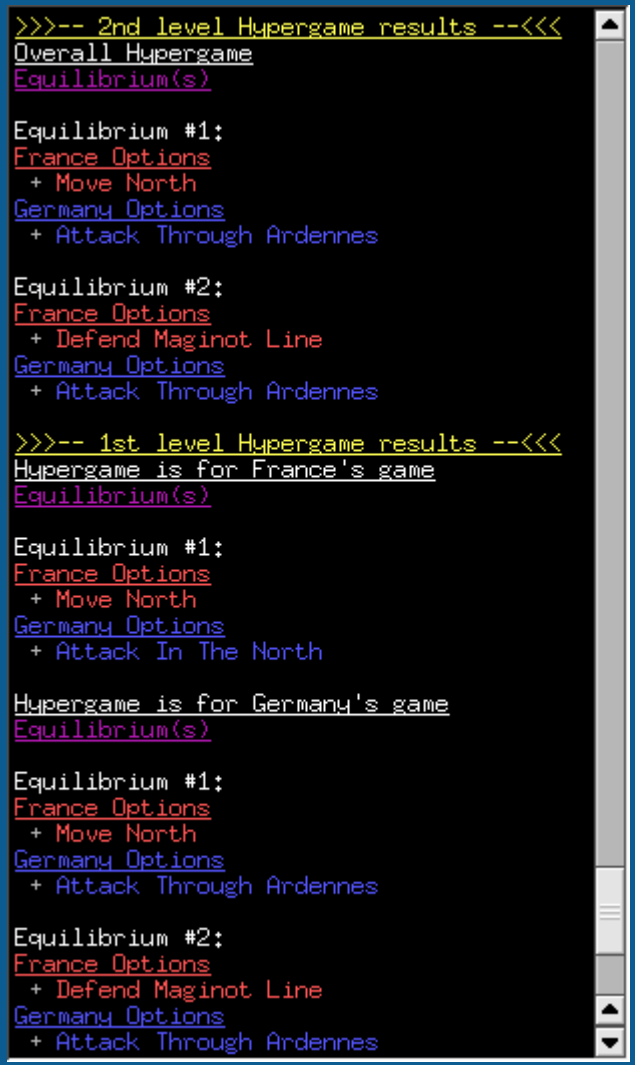
Player: USA

	Preference Vector Outcome				Stability	UIs
	#1	#2	#3	#4		
1	0	0	1	0	r	-
2	0	1	1	0	s	1
3	1	0	1	0	u	1, 2
4	1	1	1	0	u	1, 2, 3
5	0	1	0	0	r	-
6	1	0	0	0	u	5
7	1	1	0	0	u	5, 6
8	0	0	0	0	u	5, 6, 7
9	1	1	0	1	r	-
10	1	0	0	1	u	9
11	0	1	0	1	u	9, 10
12	0	0	0	1	u	9, 10, 11

Player: USSR

	Preference Vector Outcome				Stability	UIs
	#1	#2	#3	#4		
1	0	0	0	0	r	-
2	0	0	1	0	s	1
3	0	1	1	0	r	-
4	0	1	0	0	u	3
5	1	0	1	0	r	-
6	1	0	0	0	u	5
7	1	1	1	0	r	-
8	1	1	0	0	u	7
9	1	1	0	1	u	7, 8
10	1	0	0	1	u	5, 6
11	0	1	0	1	u	3, 4
12	0	0	0	1	u	1, 2

Figure 2: Stability Table for Cuban Missile Crisis


>>>-- 2nd level Hypergame results --<<<
Overall Hypergame
Equilibrium(s)
Equilibrium #1:
France Options
+ Move North
Germany Options
+ Attack Through Ardennes
Equilibrium #2:
France Options
+ Defend Maginot Line
Germany Options
+ Attack Through Ardennes
>>>-- 1st level Hypergame results --<<<
Hypergame is for France's game
Equilibrium(s)
Equilibrium #1:
France Options
+ Move North
Germany Options
+ Attack In The North
Hypergame is for Germany's game
Equilibrium(s)
Equilibrium #1:
France Options
+ Move North
Germany Options
+ Attack Through Ardennes
Equilibrium #2:
France Options
+ Defend Maginot Line
Germany Options
+ Attack Through Ardennes

Testing

Two different methods of testing where used on the analysis component of the program.

- – Convert previously analysed games into HML and analyse with HYPANT.
  - – Compare the output with those previously calculated.
- – Alter existing models to create "What If" games.
  - – Analyse altered games and see if the output seems sensible, given the input.

Program Limitations

- Hypergame models must be written in HML by a person in order to analyse them.

Future Research

- Extend parser with error recovery & sanity checking on input.
- Extend tool to allow creation of hypergames.
- Applying hypergames to Artificial Life simulations.

Questions?