# Project Summary

Modeling the placement of pieces in the world renown kanoodle puzzle.



# Propositions

* **PieceConfig(col, config)**: Piece of colour **col** is using configuration **config** (e.g., rotation).
* **PlacePiece(col, config, x, y)**: Piece of colour **col** and configuration **config** is at location (**x**,**y**).
* **PlaceColour(col, x, y)**: The colour at location (**x**,**y**) is set to **col**.

# Constraints

*List of constraint types used in the model and their (English) interpretation. You only need to provide one example for each constraint type: e.g., if you have constraints saying “cars have one colour assigned” in a car configuration setting, then you only need to show the constraints for a single car. Essentially, we want to see the pattern for all of the types of constraints, and not every constraint enumerated.*

## Only one configuration for a colour

## Don’t allow pieces out of bounds

## At most one piece can be placed at a location

## Can only place a piece in one place

## If a piece is picked for a location, then the appropriate configuration is chosen

## Every location can take on at most one colour

## If a piece is placed, then it forces the colour of the cells that make up the piece

## Initial board configuration is satisfied

# Model Exploration

*List all the ways that you have explored your model – not only the final version, but intermediate versions as well. See (C3) in the project description for ideas.*

## Visualizing the Model

Just printing the solution from Bauhaus was tough to see what was going on, so we used the **tabulate** library to illustrate where pieces go. For example:

Table

Description automatically generated with medium confidence

Brackets show the colour of the square, and capital shows the piece colour placed.

Ideas on how to improve this for the final:

* Move to shaded colour background for the pieces that are placed.
* ..

## Fixing the Overlap

At one point, we were facing an issue where pieces would be overlapping. An example:

A picture containing table

Description automatically generated

To fix this, we looked into adding constraints that prevented overlap. This ultimate lead to a new proposition (the third one discussed above) that says what colour a location should be.

# Jape Proofs

Because the model is too large to fit everything, we will prove sequents over a far reduced size of kanoodle board. 2x2 or 3x3 are the most likely dimensions. Also, we will only consider pairs of pieces.

To discuss the ideas, consider blue and red pieces for this 3x3 board:

Table

Description automatically generated with low confidence

Ideas for what we might want to prove in Jape:

* Placing a blue piece facing left (on the far right side) means that only one of two possible red placements are possible.
* Placing a piece in a particular location (e.g., red in the top left) will force another square to be red because of where the piece goes.
* I can deduce anything (anything!) if I end up putting two red pieces down.

# First-Order Extension

*Describe how you might extend your model to a predicate logic setting, including how both the propositions and constraints would be updated.* ***There is no need to implement this extension!***

# Requested Feedback

1. I’m really unsure about these Jape proof ideas. Too hard? Too easy? Help!
2. Our pieces still overlap for some reason. How do we figure this out?
3. Any ideas on visualizing things better, would be more than welcome.

# Useful Notation

*Feel free to copy/paste the symbols here and remove this section before submitting.*