INF2D Reasoning and Agents Coursework 1

CSPs and Search (Due 03/03)

Learning Objectives

- Gain practical experience defining and solving CSP problems
- Better understand the A* search algorithm
- Understand the limitations of various search heuristics and develop the ability to choose appropriate heuristics based on domain knowledge
- Understand the difference between search and optimization

Coursework Structure

- No more manual working out or Haskell!!!
- You are asked to solve two problems based on ‘real world’ scenarios
- You are given a Jupyter notebook with blanks to fill in

Coursework Structure cont…

- The coursework has two parts:
  - Part A: CSPs
  - Part B: Search
- Give yourself enough time
- CW2 is out before CW1 deadline so plan accordingly
- CW1 will be due 03/03 at 12pm

Part A: CSP (50%)

You will be asked to find a valid deck of Dobble cards for an arbitrary number of symbols.

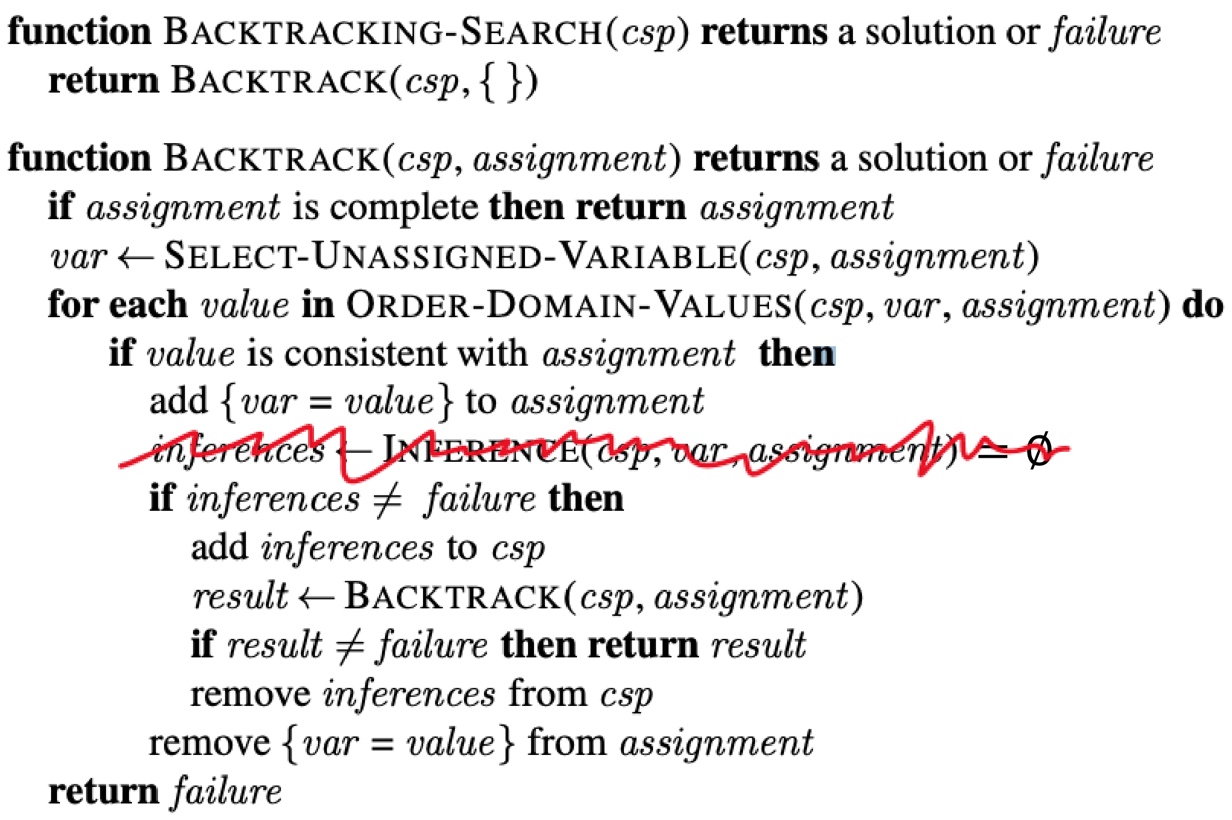
What is Dobble?

- A card game with 55 cards in the deck
- Each card has 8 symbols on it
- For the game to work each pair of cards must share exactly one symbol between them

Part A: CSP (50%)

1. Model the behaviour of the cards and the deck
2. Formalise the problem
3. Solve with backtracking search
4. Investigate the impact of parameters on the search space and runtime

Backtracking Algorithm



```
function BACKTRACKING-SEARCH(csp) returns a solution or failure
  return BACKTRACK(csp, { })

function BACKTRACK(csp, assignment) returns a solution or failure
  if assignment is complete then return assignment
  var ← SELECT-UNASSIGNED-VARIABLE(csp, assignment)
  for each value in ORDER-DOMAIN-VALUES(csp, var, assignment) do
     if value is consistent with assignment then
        add {var = value} to assignment
        ~~inferences ← INFERENCE(csp, var, assignment)~~ = ∅
        if inferences ≠ failure then
           add inferences to csp
           result ← BACKTRACK(csp, assignment)
           if result ≠ failure then return result
           remove inferences from csp
        remove {var = value} from assignment
  return failure
```

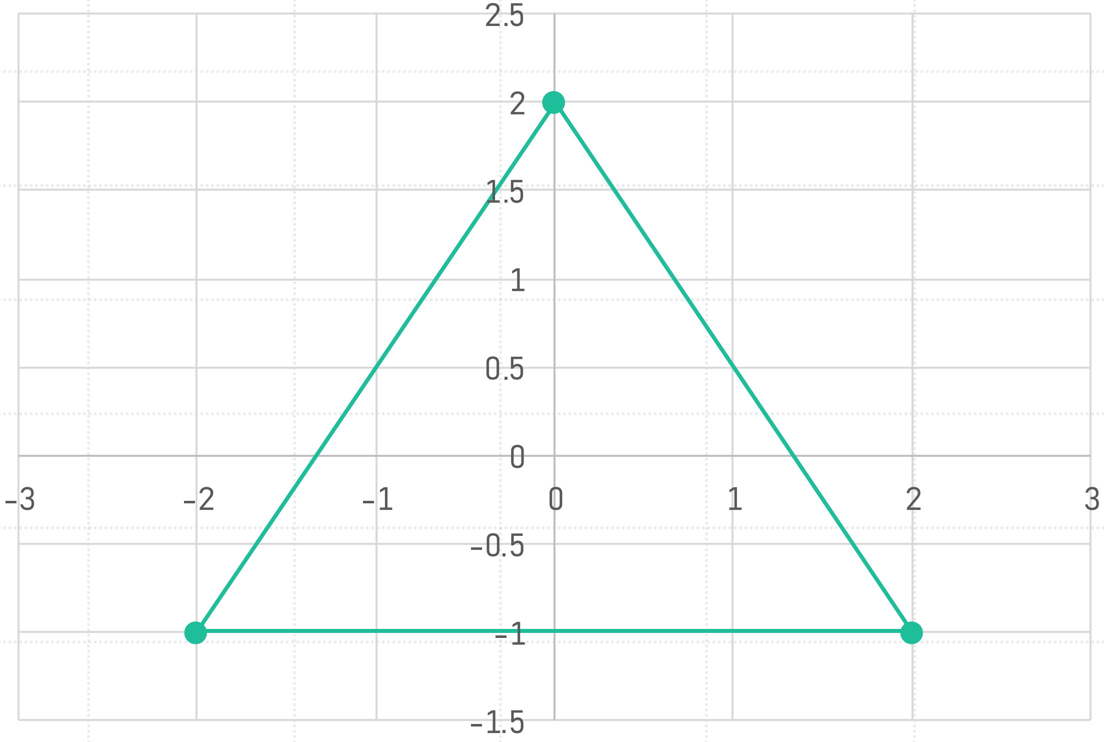
See Russell and Norvig chapter 6 for details

Part B: Search (50%)

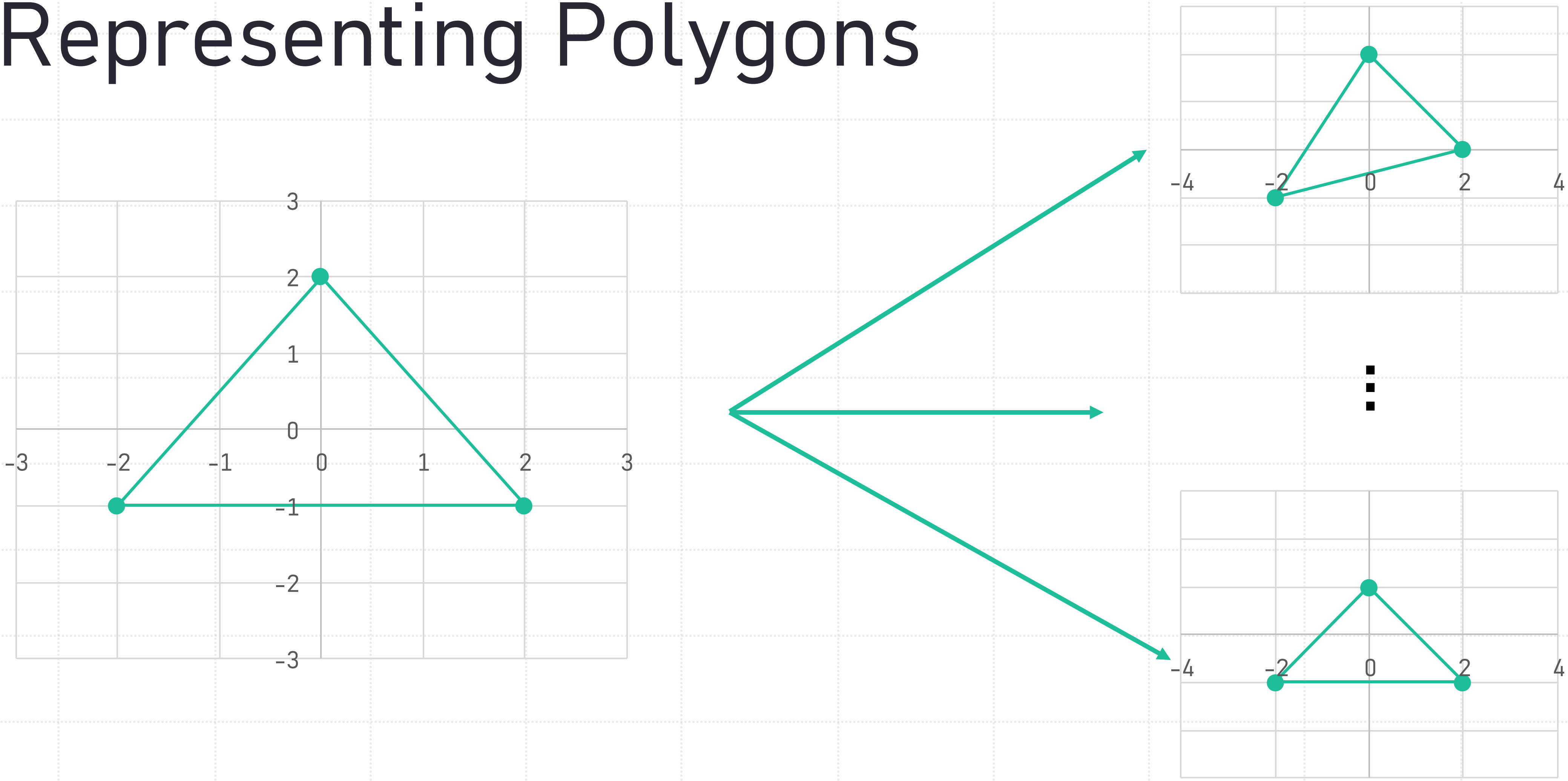
- This is an A* Search problem with a twist
- You are working on a new animation software
- Your job is to write an algorithm that “smoothly” transitions from one polygon to another

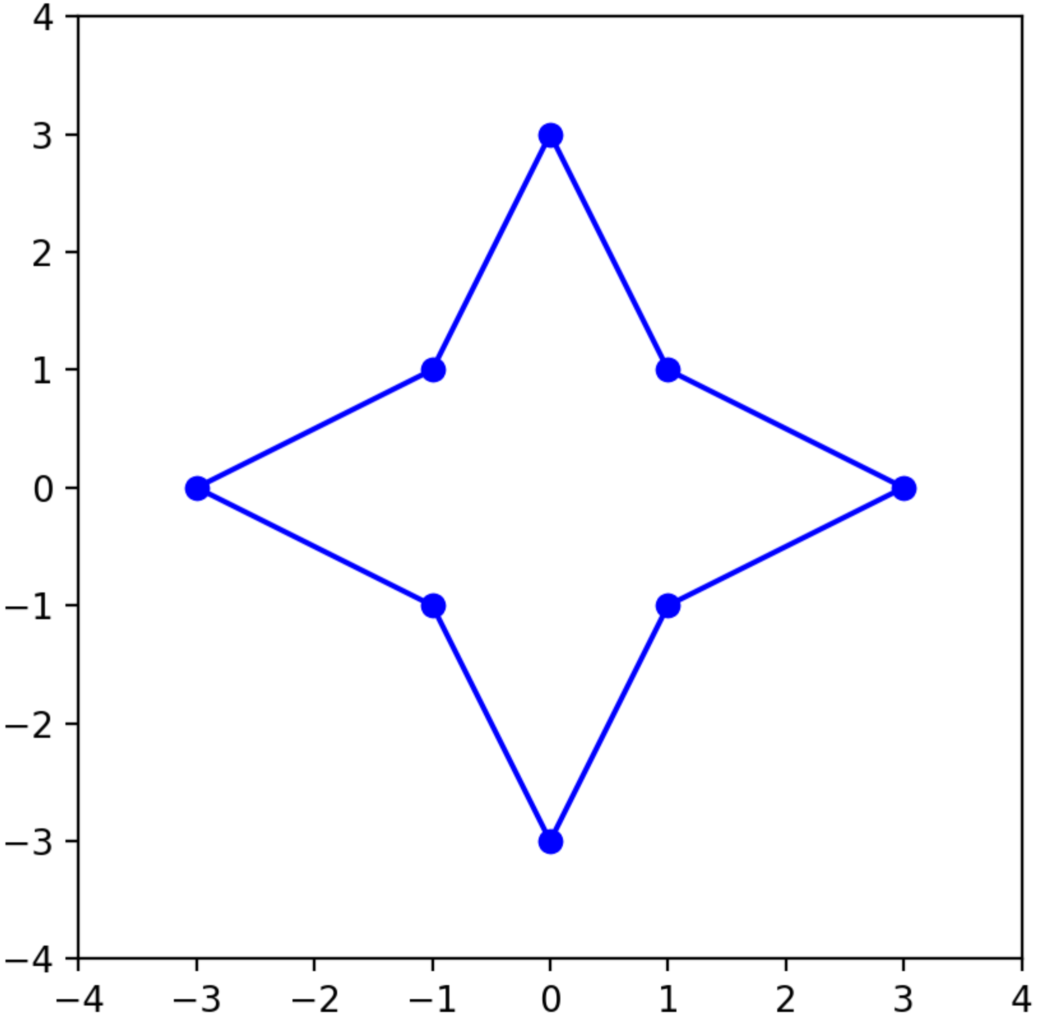
Representing Polygons

- Polygons are represented as an ordered list of points
- [(0, 2), (2, -1), (-2, -1)] gives you the triangle you see here.

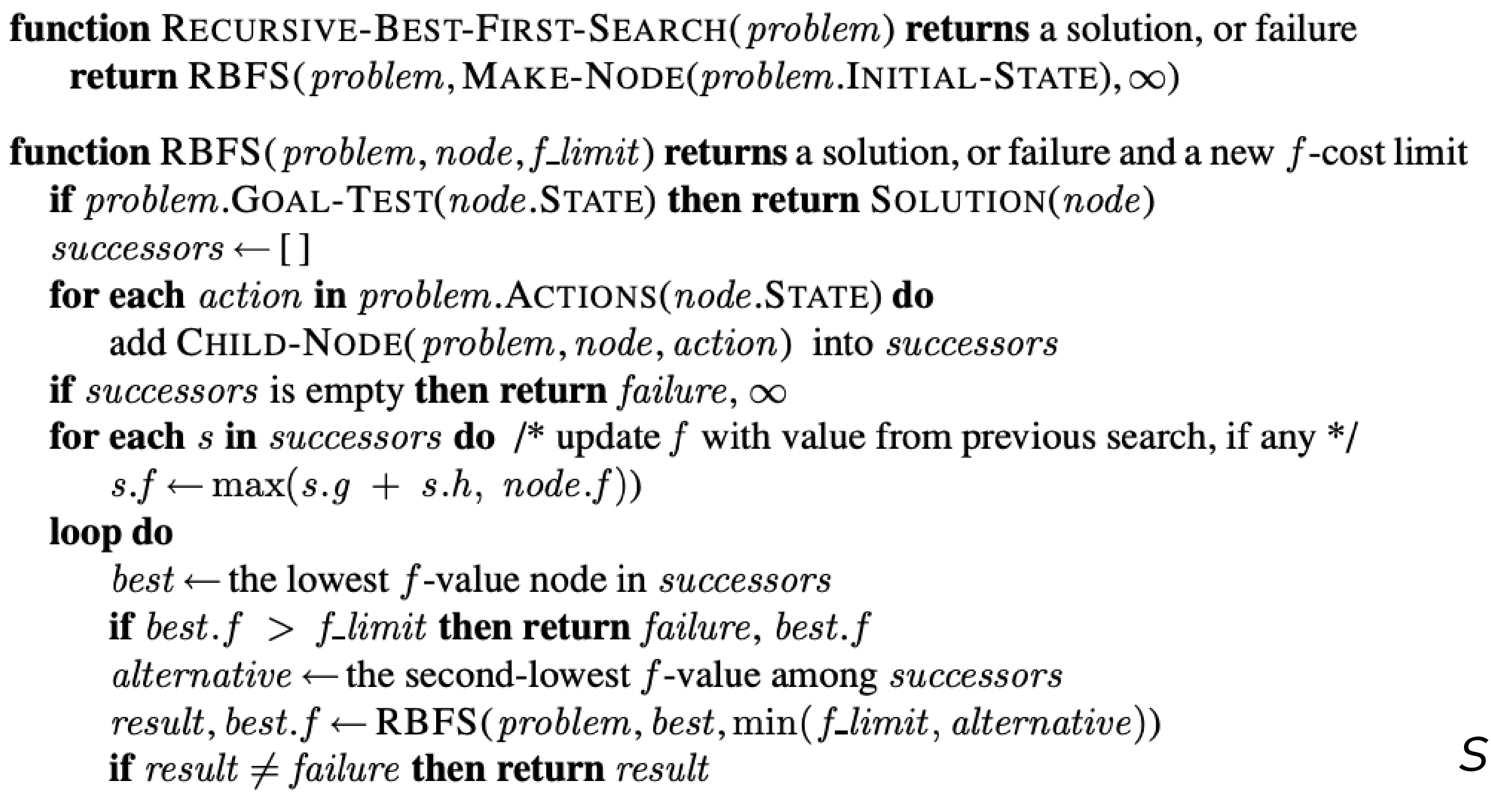


Representing Polygons




4
3
2
1
0
−1
−2
−3
−4
−4
−3
−2
−1
0
1
2
3
4

A* Search Algorithm

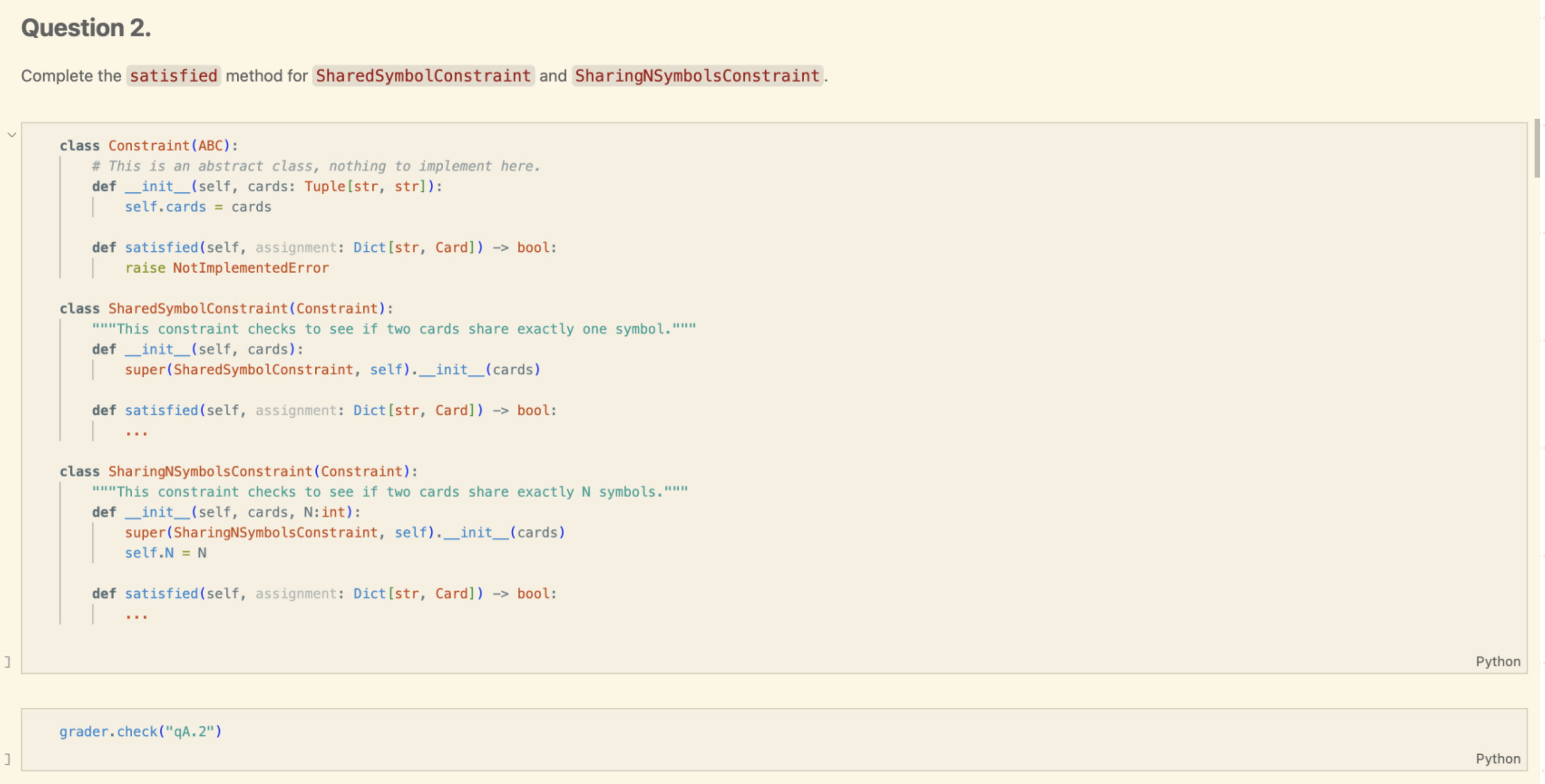


```
function RECURSIVE-BEST-FIRST-SEARCH(problem) returns a solution, or failure
    return RBFS(problem, MAKE-NODE(problem.INITIAL-STATE), $\infty$)

function RBFS(problem, node, f_limit) returns a solution, or failure and a new f-cost limit
  if problem.GOAL-TEST(node.STATE) then return SOLUTION(node)
  successors ← [ ]
  for each action in problem.ACTIONS(node.STATE) do
      add CHILD-NODE(problem, node, action) into successors
  if successors is empty then return failure, $\infty$
  for each s in successors do /* update f with value from previous search, if any */
      $s.f \leftarrow \max(s.g + s.h,\ node.f)$
  loop do
      best ← the lowest f-value node in successors
      if $best.f > f\_limit$ then return failure, best.f
      alternative ← the second-lowest f-value among successors
      result, best.f ← RBFS(problem, best, min(f_limit, alternative))
      if $result \neq failure$ then return result
```

See Russell and Norvig, 3.5 for details

Submission Instructions

Notebook Format



Question 2.

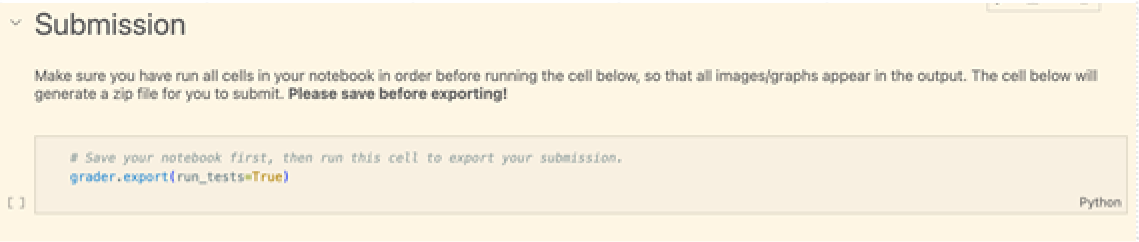
Complete the `satisfied` method for `SharedSymbolConstraint` and `SharingNSymbolsConstraint`.

```python
class Constraint(ABC):
    # This is an abstract class, nothing to implement here.
    def __init__(self, cards: Tuple[str, str]):
        self.cards = cards

    def satisfied(self, assignment: Dict[str, Card]) -> bool:
        raise NotImplementedError

class SharedSymbolConstraint(Constraint):
    """This constraint checks to see if two cards share exactly one symbol."""
    def __init__(self, cards):
        super(SharedSymbolConstraint, self).__init__(cards)

    def satisfied(self, assignment: Dict[str, Card]) -> bool:
        ...

class SharingNSymbolsConstraint(Constraint):
    """This constraint checks to see if two cards share exactly N symbols."""
    def __init__(self, cards, N:int):
        super(SharingNSymbolsConstraint, self).__init__(cards)
        self.N = N

    def satisfied(self, assignment: Dict[str, Card]) -> bool:
        ...
```

```python
grader.check("qA.2")
```

Exporting Your Notebook

1. Restart the kernel
2. Run all cells
3. If you are happy with the outputs save the notebook
4. Run “grader.export()”
5. Submit the generated zip file to Gradescope



Submission

Make sure you have run all cells in your notebook in order before running the cell below, so that all images/graphs appear in the output. The cell below will generate a zip file for you to submit. **Please save before exporting!**

```
# Save your notebook first, then run this cell to export your submission.
grader.export(run_tests=True)
```

Academic Misconduct

- Academic misconduct is any type of cheating that occurs in relation to a formal academic exercise.
- Includes plagiarism, collusion, falsification, deceit, cheating and personation.
- The University takes all reported incidences of academic misconduct seriously and seeks to ensure that they are dealt with efficiently and appropriately.
- **Help each other out, give each other hints and guidance but don’t share answers.**

Getting Support

Labs

- Weeks 4-6 in AT 6.06 [ROOM CHANGE]
- Provide support and make sure you're on the right track
- Demonstrators are able to guide you but won't give you answers

Piazza

- If you have a question someone else probably has it too
- Make sure to not give away solutions in your posts
- If you're not sure, hide it

Me

- I am here if you really feel stuck
- Email me @ ameer.saadat@ed.ac.uk
- Feedback on the coursework is also appreciated ☺
- Basically, leave the lecturers alone

Questions?