Lecture 10 (Constraint Satisfaction)

The algorithms that we have seen do not explicitly look at the structure and find specific "patterns"

1 Constraint Satisfaction Problems (CSPs)

1. $\{X_i\}_{i=1}^n$ are variables that are given values $\{d_i\}_{i=1}^n$ from domain D
2. Finding solution involves assigning values to $\{X_i\}_{i=1}^n$ such that it is consistent (satisfies all constraints)

2 Constraint Graph

1. Binary constraint statisfaction problem (each edge relates atmost two variables)
2. Nodes are variables and edges show constraints
3. Generic CSP solvers use the graph structure to speed up search

3 Types of Constraints

1. Boolean - permitted or not permitted assignment
2. Preferential - some assignments better than others
3. Unary/binary/n-ary

4 Solving CSP

1. Initial state: empty assignment {}
2. Successor function: assign a value to an unassigned variable
3. Goal test: assignment is complete and satisfies all constraints
4. To improve complexity, stop exploring once the partial assignment is unsatisfiable

4.1 Backtracking Search

```
def RecursiveBacktracking(assignment, csp):
    if assignment is complete:
        return assignment
```

```
var = SelectUnassignedVariable(Variables[csp], assignment, csp)
for each value in OrderDomainValues(var, assignment, csp):
    if value is consistent with assignment given Constraints[csp]:
        add {var = value} to assignment
        result = RecursiveBacktracking(assignment, csp)
        if result != failure:
            return result
        remove {var = value} from assignment
return failure
```

4.2 Improving Efficiency

1. Choose the most constrained variable at every step
2. To resolve ties in above criteria, choose the vertex with largest degree