Section 4

CSP'S AND LOCAL SEARCH

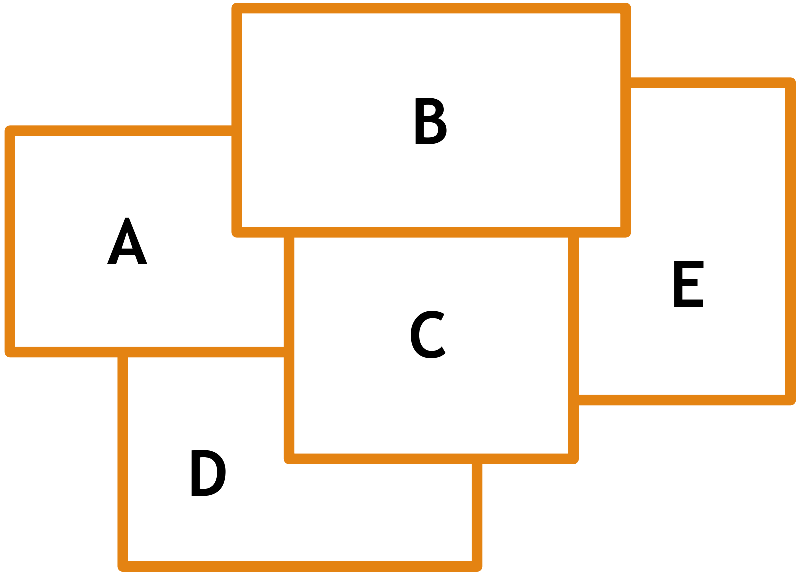
CSPs and Local Search

Last week in CSPs:
- variables, domains, constraints
- backtracking

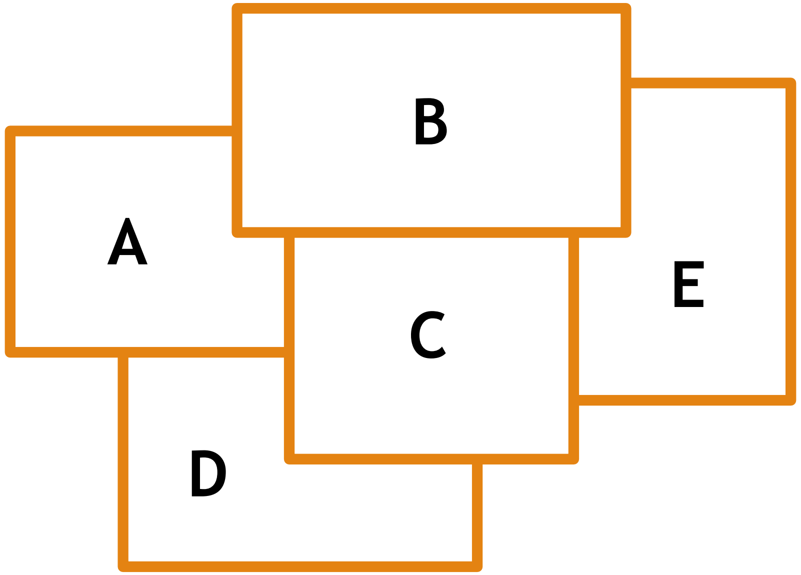
This week in CSPs:
- ordering
- forward checking
- arc consistency

Also, in local search:
- optimization and convexity
- hill climbing
- beam search
- simulated annealing
- gradient descent
- genetic algorithms

Constraint Satisfaction Problems

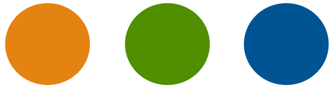


Constraint Satisfaction Problems



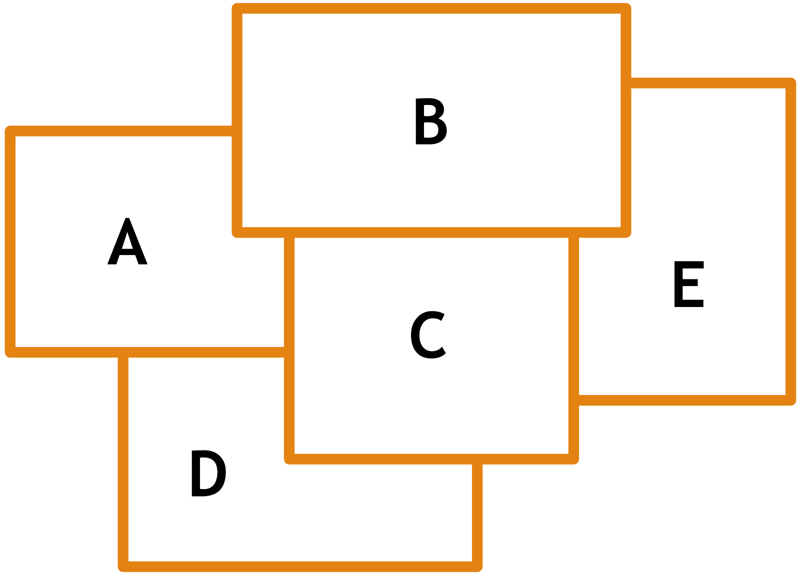
Variables: A, B, C, D, E

Values:



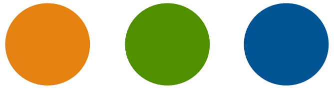
Constraints:

Constraint Satisfaction Problems



Variables: A, B, C, D, E

Values:



Constraints:

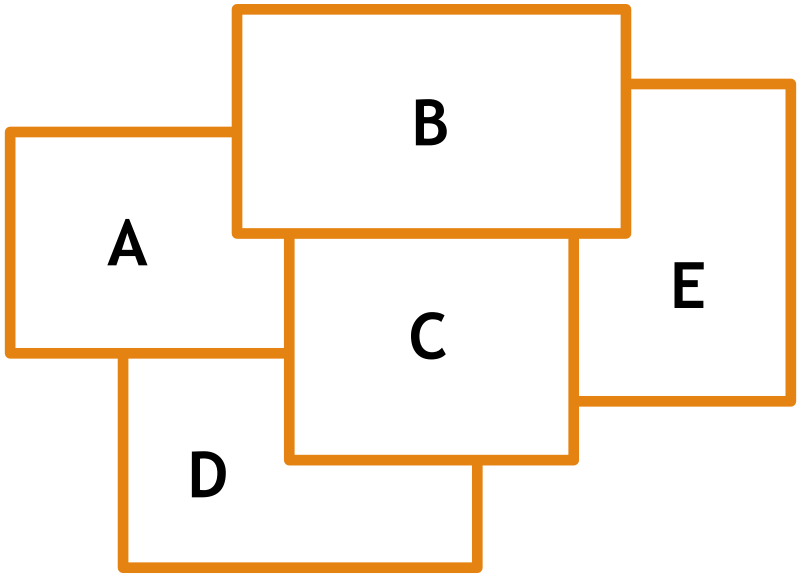
A != B
A != C
A != D
B != E
B != C
C != E
D != C

Constraint Satisfaction Problems

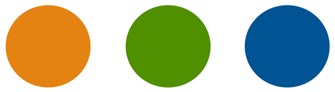
Most naive search tree:
branching factor of 15

Fixed ordering search tree:
branching factor of 3

What is a better way to order?



Variables: A, B, C, D, E

Values: 

Constraints:

A != B
A != C
A != D
B != E
B != C
C != E
D != C

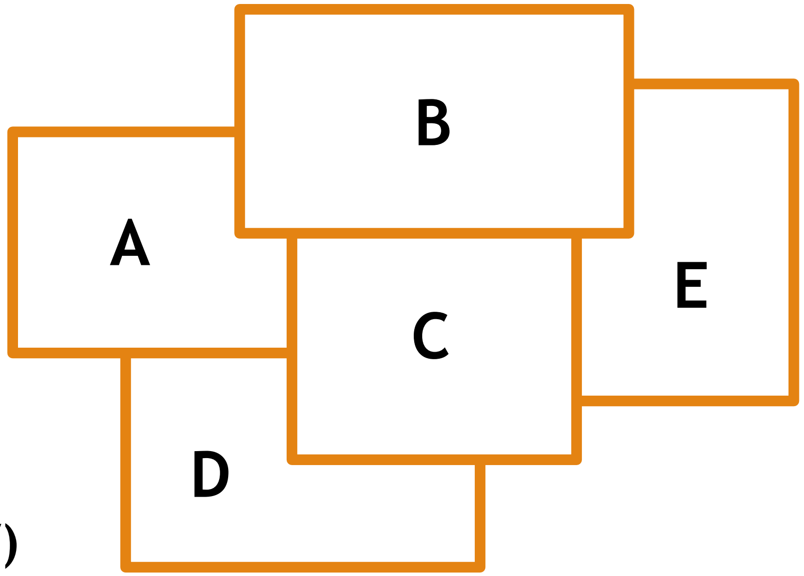
Constraint Satisfaction Problems

Most naive search tree:
branching factor of 15

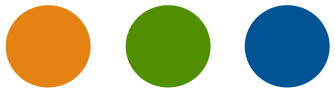
Fixed ordering search tree:
branching factor of 3

What is a better way to order?

Heuristics:
- **Minimum remaining values (MRV)**
- **Fewest constraints (degree)**
- **Least constraining value**

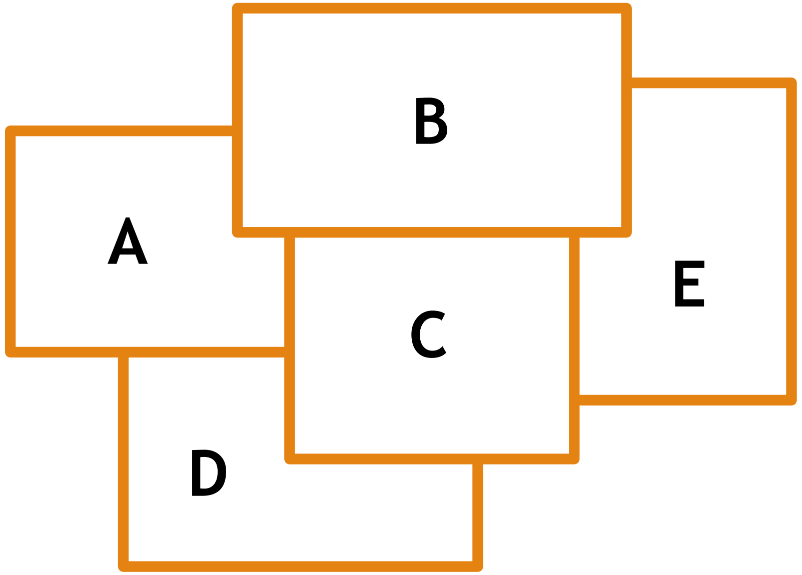


Variables: A, B, C, D, E

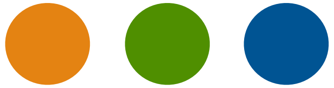
Values: 

Constraints:
A != B
A != C
A != D
B != E
B != C
C != E
D != C

Constraint Satisfaction Problems

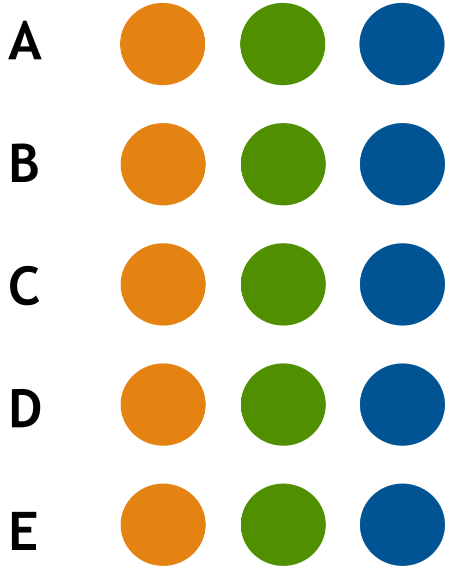


Variables: A, B, C, D, E

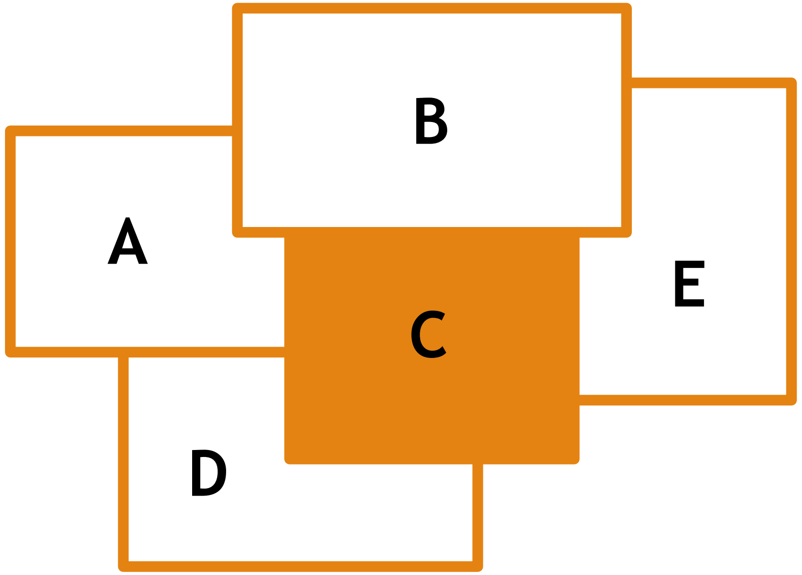
Values: 

Constraints:

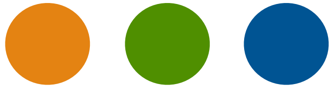
A != B
A != C
A != D
B != E
B != C
C != E
D != C



Constraint Satisfaction Problems

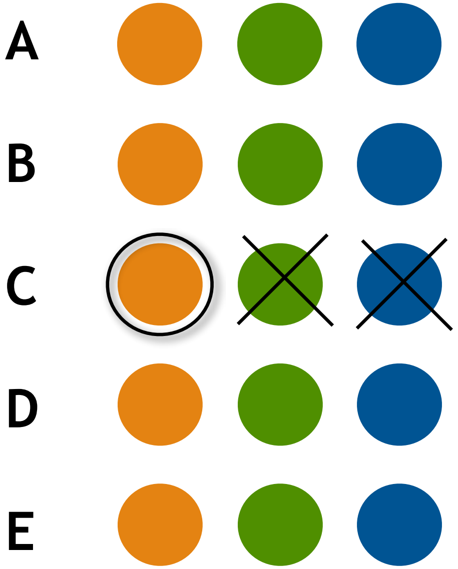


<u>Variables:</u> A, B, C, D, E

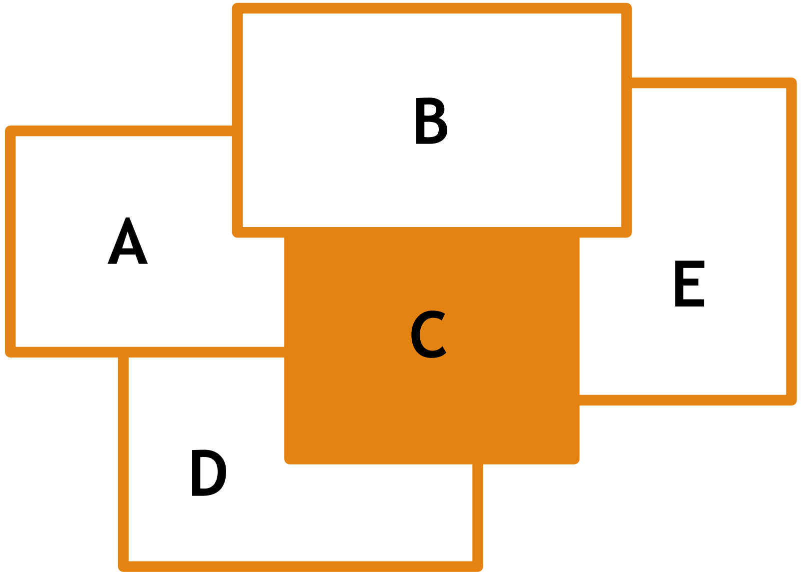
<u>Values:</u> 

<u>Constraints:</u>

A != B
A != C
A != D
B != E
B != C
C != E
D != C

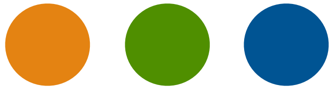


Constraint Satisfaction Problems



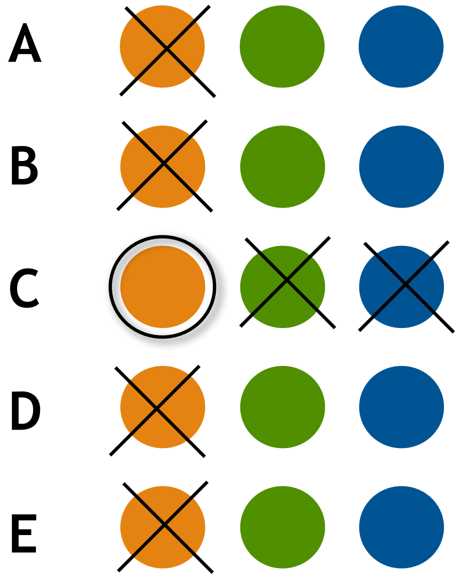
Variables: A, B, C, D, E

Values:

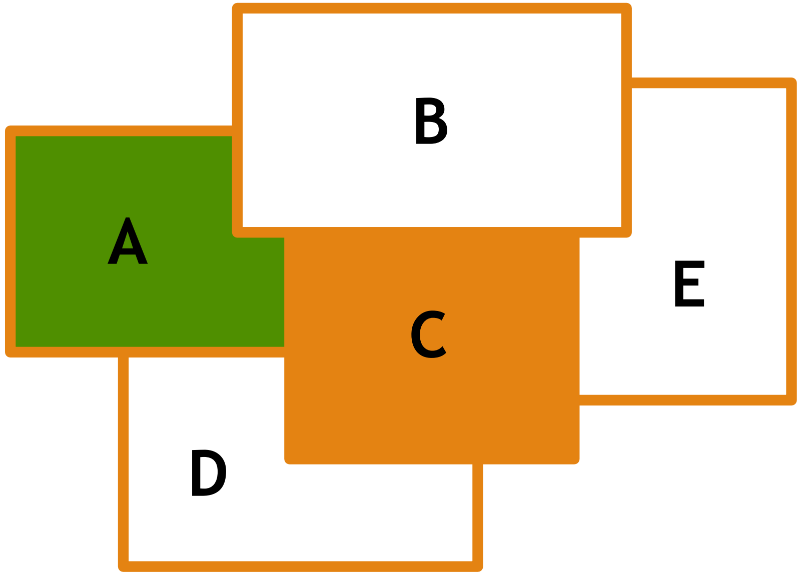


Constraints:

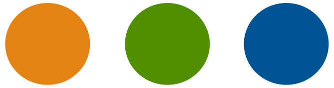
A != B
A != C
A != D
B != E
B != C
C != E
D != C



Constraint Satisfaction Problems

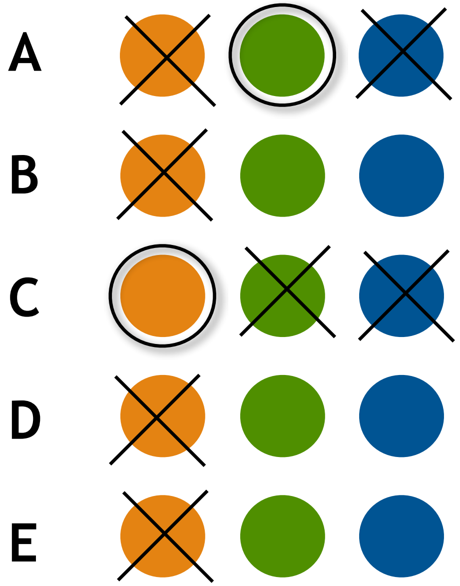


Variables: A, B, C, D, E

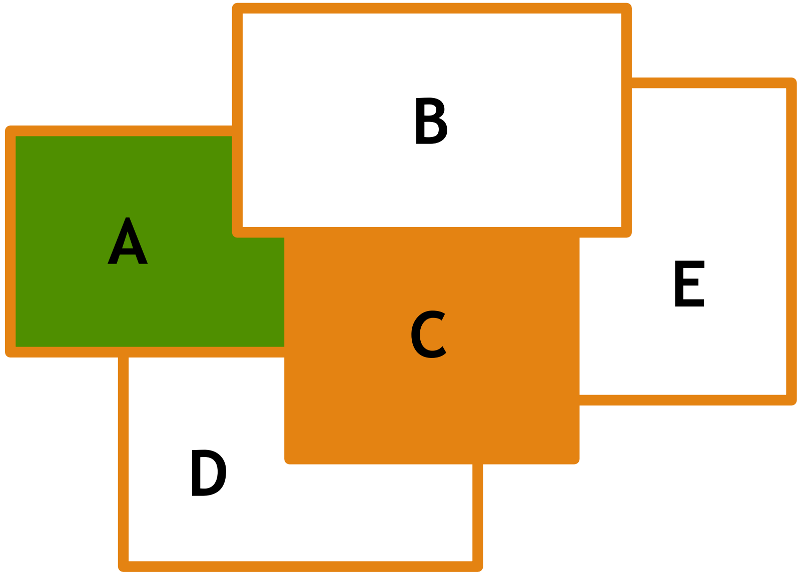
Values: 

Constraints:

A != B
A != C
A != D
B != E
B != C
C != E
D != C



Constraint Satisfaction Problems

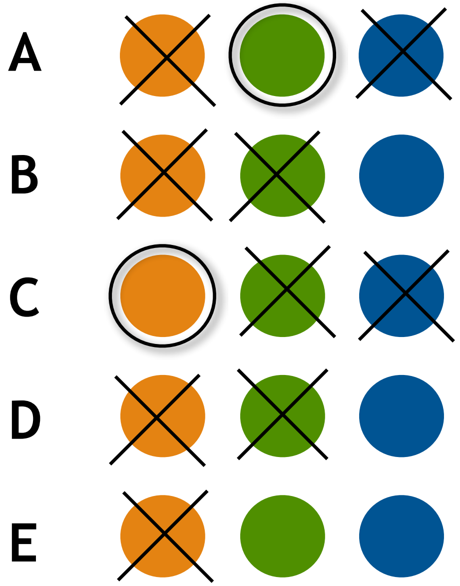


Variables: A, B, C, D, E

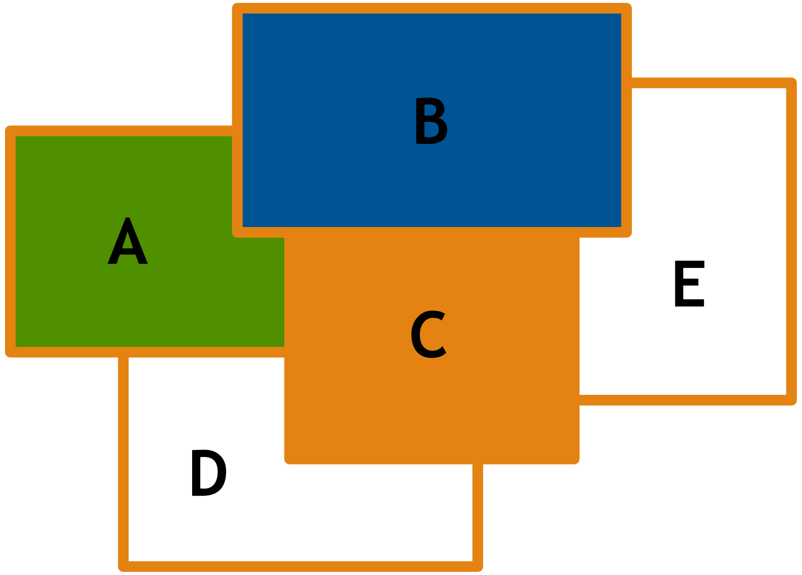
Values:

Constraints:

A != B
A != C
A != D
B != E
B != C
C != E
D != C

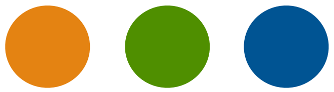


Constraint Satisfaction Problems



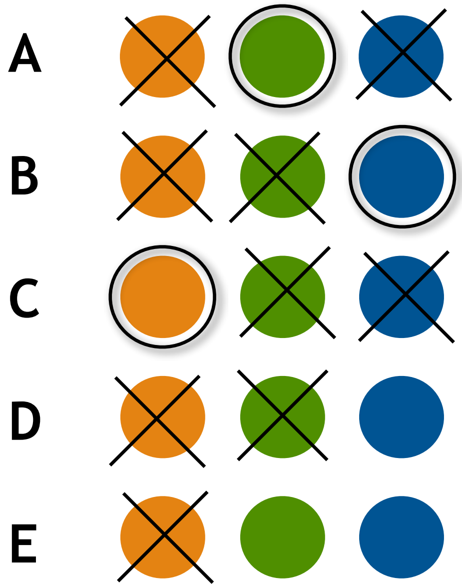
Variables: A, B, C, D, E

Values:

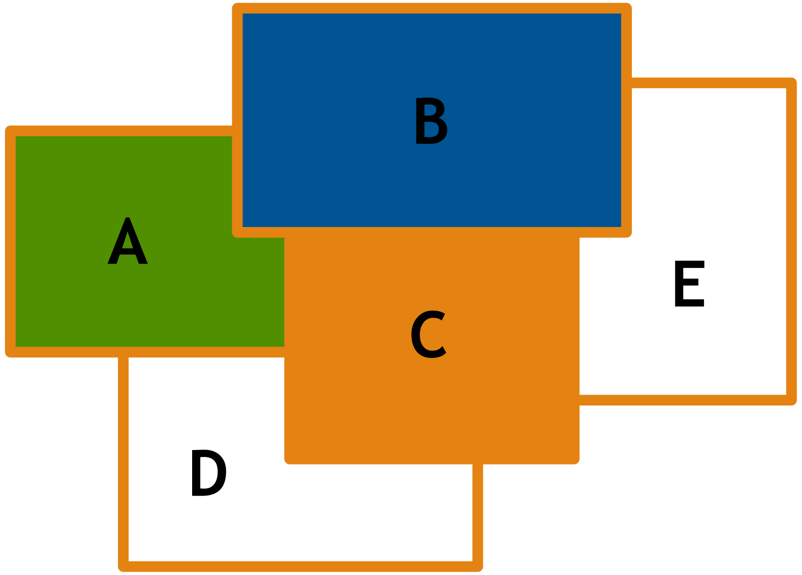


Constraints:

A != B
A != C
A != D
B != E
B != C
C != E
D != C



Constraint Satisfaction Problems

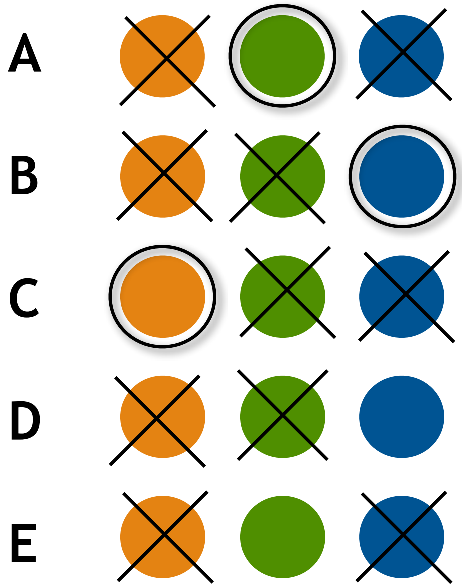


Variables: A, B, C, D, E

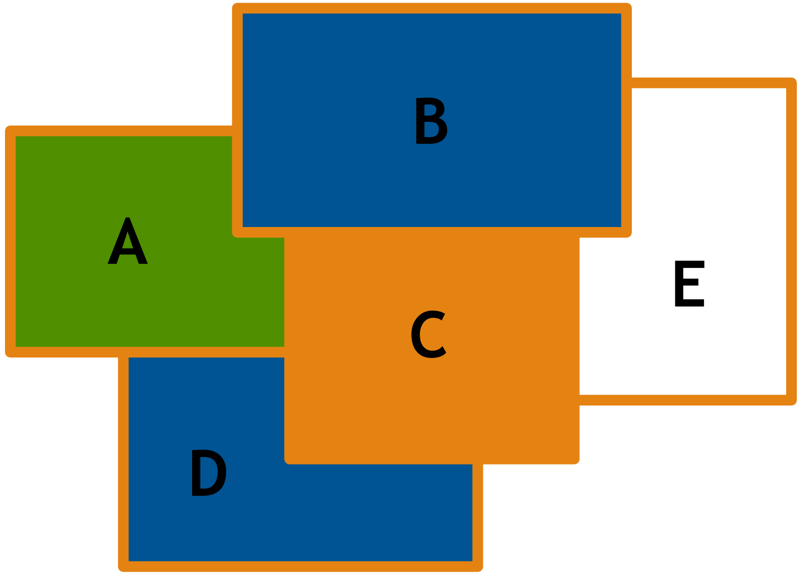
Values:

Constraints:

A != B
A != C
A != D
B != E
B != C
C != E
D != C



Constraint Satisfaction Problems

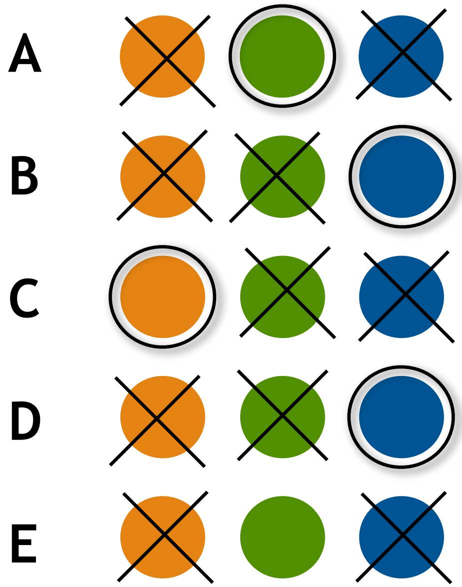


<u>Variables:</u> A, B, C, D, E

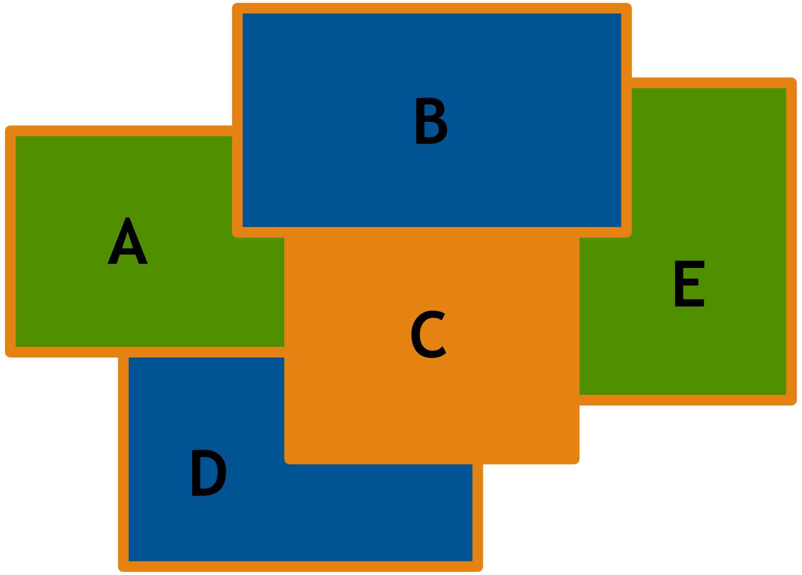
<u>Values:</u>

<u>Constraints:</u>

A != B
A != C
A != D
B != E
B != C
C != E
D != C

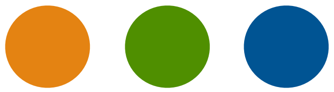


Constraint Satisfaction Problems



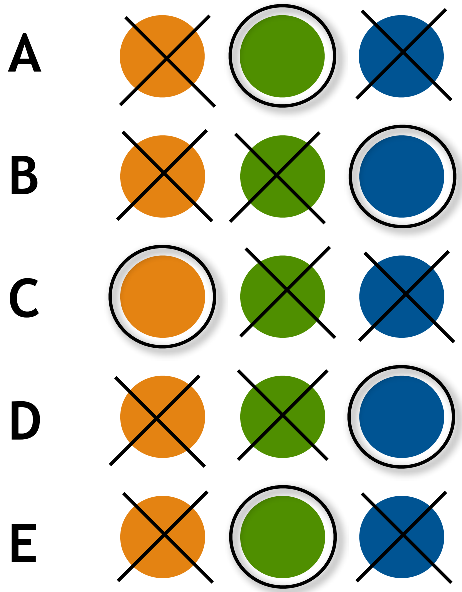
Variables: A, B, C, D, E

Values:



Constraints:

A != B
A != C
A != D
B != E
B != C
C != E
D != C



Optimization

Hill climbing

Beam search

Simulated annealing

Genetic algorithms

Gradient descent

Optimization

Hill climbing
Given a state, generate all successors and choose the best one. If none are better, return current state.

Beam search

Simulated annealing

Genetic algorithms

Gradient descent

Optimization

Hill climbing
Given a state, generate all successors and choose the best one. If none are better, return current state.

Beam search
Choose *k* initial states. For all states, expand all successors. Among the generated states, choose the best k and continue until convergence.

Simulated annealing

Genetic algorithms

Gradient descent

Optimization

Hill climbing
Given a state, generate all successors and choose the best one. If none are better, return current state.

Beam search
Choose *k* initial states. For all states, expand all successors. Among the generated states, choose the best k and continue until convergence.

Simulated annealing
At each iteration, choose a random successor. Accept the successor with a “temperature”-dependent probability, which decreases over time.

Genetic algorithms

Gradient descent

Optimization

Hill climbing
Given a state, generate all successors and choose the best one. If none are better, return current state.

Beam search
Choose *k* initial states. For all states, expand all successors. Among the generated states, choose the best k and continue until convergence.

Simulated annealing
At each iteration, choose a random successor. Accept the successor with a “temperature”-dependent probability, which decreases over time.

Genetic algorithms
At each iteration, generate (and mutate) successors from the “best” sample in the previous iteration.

Gradient descent

Optimization

Hill climbing
Given a state, generate all successors and choose the best one. If none are better, return current state.

Beam search
Choose *k* initial states. For all states, expand all successors. Among the generated states, choose the best k and continue until convergence.

Simulated annealing
At each iteration, choose a random successor. Accept the successor with a “temperature”-dependent probability, which decreases over time.

Genetic algorithms
At each iteration, generate (and mutate) successors from the “best” sample in the previous iteration.

Gradient descent
Calculate and repeatedly step down the gradient (for continuous functions).