COMP3211 Tutorial 5: Satisfiability

Fengming ZHU

Mar. 11&14, 2024

Department of CSE
HKUST

Outline

Boolean Satisfiability Problems (SAT)

Preliminaries

Boolean Satisfiability Problem (SAT):

$$\neg X_1 \wedge X_2 \vee (X_3 \wedge X_4) \wedge \neg X_3 \vee \cdots$$

Conjunctive normal form (CNF):

$$(\neg X_1 \vee X_2) \wedge (X_1 \vee X_3 \vee X_4) \wedge (\neg X_3) \wedge \cdots$$

Literals | Clauses | Variables

Solving SAT as Search

$\mathcal{X} = \{x_1, x_2, x_3\}$ $\{x_1 = T\}$. $\{x_1 = T, x_2 = F\}$

n

Reduce to SEARCH formulation: $2n$. $3n$. n^2 n^3

- States: any partial assignment of the variables in $\mathcal{X}$. 2^n. 3^n

Solving SAT as Search

Reduce to SEARCH formulation:

- States: any partial assignment of the variables in $\mathcal{X}$.
- Initial state: empty assignment $\{\}$.

Solving SAT as Search

Reduce to SEARCH formulation:

- States: any partial assignment of the variables in $\mathcal{X}$.
- Initial state: empty assignment $\{\}$.
- Operators: $let(x, v)$ that assigns variable x value v (1/0). This operator is applicable in state s if and only if the variable x does not have a value in s.

Solving SAT as Search

Reduce to SEARCH formulation:

- States: any partial assignment of the variables in $\mathcal{X}$.
- Initial state: empty assignment $\{\}$.
- Operators: $let(x, v)$ that assigns variable x value v (1/0). This operator is applicable in state s if and only if the variable x does not have a value in s.
- Operator cost: 1 for each operator.

Solving SAT as Search

Reduce to SEARCH formulation:

- States: any partial assignment of the variables in $\mathcal{X}$.
- Initial state: empty assignment $\{\}$.
- Operators: $let(x, v)$ that assigns variable x value v (1/0). This operator is applicable in state s if and only if the variable x does not have a value in s.
- Operator cost: 1 for each operator.
- Goal states: those that make all clauses in $\mathcal{C}$ true.

Solving SAT as Search

n SAT → NPC → exp(n)

m Search → P → poly(m)

$m \sim exp(n)$

$\sim poly(exp(n))$

$= exp(n)$

Constraint Satisfiability Problems (CSP)

Formulation

Notations:

- A set of Variables $\mathcal{X}$
- Each variable $x_i \in \mathcal{X}$ is associated with a set of domain values $\mathcal{V}_i$
- A set of constraints (conditions) on variables, could be unitary, binary or else.
- A solution is an assignment where all variables get assigned and all constraints are satisfied.

Constraints Propagation

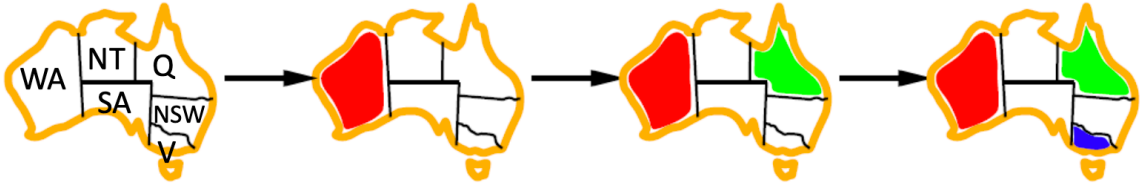
How to do search:

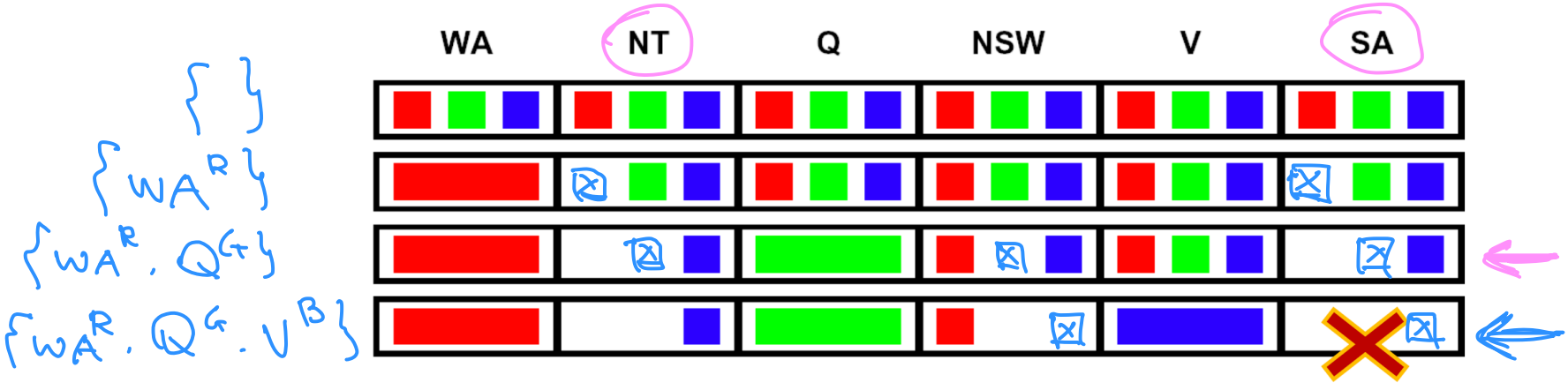
- Naive way: backtracking search (depth-first search)

Constraints Propagation

How to do search:

- Naive way: backtracking search (depth-first search)
- Make use of constraints – constraints propagation

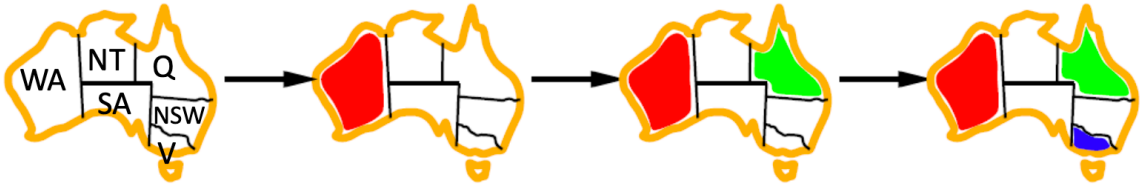




Constraints Propagation

How to do search:

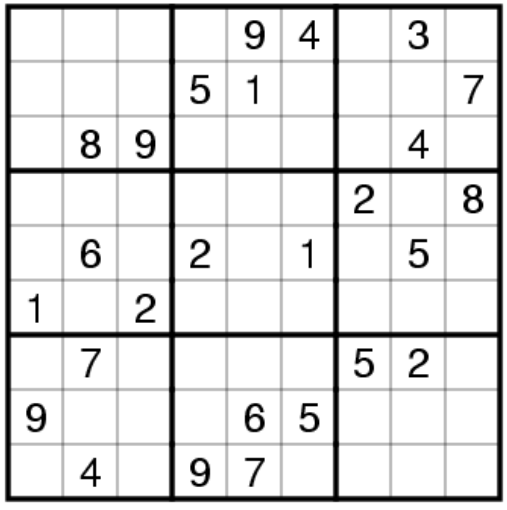
- Naive way: backtracking search (depth-first search)
- Make use of constraints – constraints propagation



Edge

- Furthermore: arc consistency

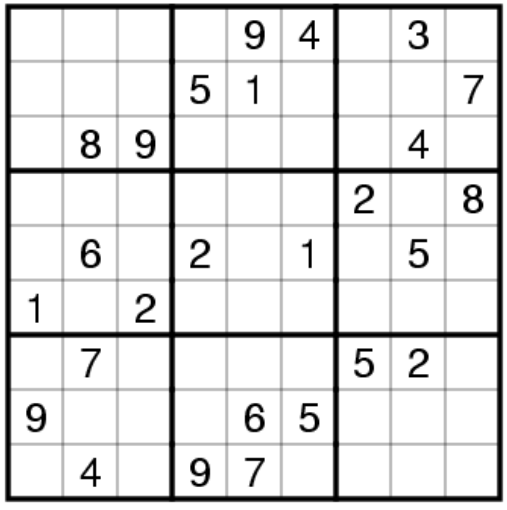
Example: Sudoku as CSP



As a CSP:

- Variables: X_{ij}, for ungiven $i, j \in [1..9]$

Example: Sudoku as CSP

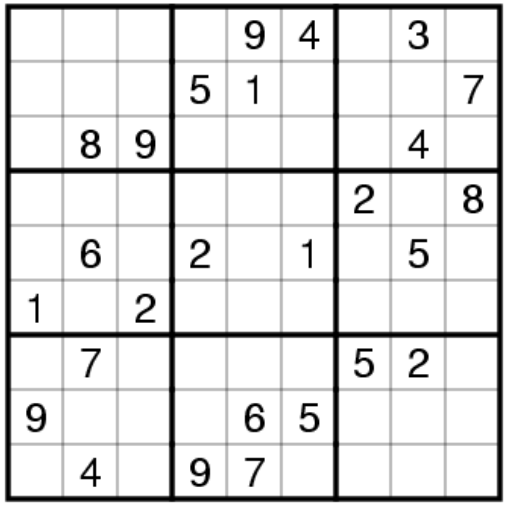


				9	4		3	
			5	1				7
	8	9					4	
						2		8
	6		2		1		5	
1		2						
	7					5	2	
9				6	5			
	4		9	7				

As a CSP:

- Variables: X_{ij}, for ungiven $i, j \in [1..9]$
- Domain values: $X_{ij} \in [1..9]$ for all ungiven $i, j \in [1..9]$

Example: Sudoku as CSP



				9	4		3	
			5	1				7
	8	9					4	
						2		8
	6		2		1		5	
1		2						
	7					5	2	
9				6	5			
	4		9	7				

As a CSP:

- Variables: X_{ij}, for ungiven $i, j \in [1..9]$
- Domain values: $X_{ij} \in [1..9]$ for all ungiven $i, j \in [1..9]$
- Constraints: each row, column and square has DISTICNT $[1..9]$

Example 2: Sudoku as CSP (cont'd)

				9	4		3	
			5	1				7
	8	9					4	
						2		8
	6		2		1		5	
1		2						
	7					5	2	
9				6	5			
	4		9	7				

```
[[7, 1, 5, 8, 9, 4, 6, 3, 2],
 [2, 3, 4, 5, 1, 6, 8, 9, 7],
 [6, 8, 9, 7, 2, 3, 1, 4, 5],
 [4, 9, 3, 6, 5, 7, 2, 1, 8],
 [8, 6, 7, 2, 3, 1, 9, 5, 4],
 [1, 5, 2, 4, 8, 9, 7, 6, 3],
 [3, 7, 6, 1, 4, 8, 5, 2, 9],
 [9, 2, 8, 3, 6, 5, 4, 7, 1],
 [5, 4, 1, 9, 7, 2, 3, 8, 6]]
```

Example 2: Sudoku as CSP (cont'd)

				9	4		3	
			5	1				7
	8	9					4	
						2		8
	6		2		1		5	
1		2						
	7					5	2	
9				6	5			
	4		9	7				

```
[[7, 1, 5, 8, 9, 4, 6, 3, 2],
 [2, 3, 4, 5, 1, 6, 8, 9, 7],
 [6, 8, 9, 7, 2, 3, 1, 4, 5],
 [4, 9, 3, 6, 5, 7, 2, 1, 8],
 [8, 6, 7, 2, 3, 1, 9, 5, 4],
 [1, 5, 2, 4, 8, 9, 7, 6, 3],
 [3, 7, 6, 1, 4, 8, 5, 2, 9],
 [9, 2, 8, 3, 6, 5, 4, 7, 1],
 [5, 4, 1, 9, 7, 2, 3, 8, 6]]
```

More applications:

- Course scheduling
- Bin packing
- Housing allocation

Thanks!