Announcements

- Solution to Assignment 2 is posted
- Assignment 3 is available

Review: Searching

- Generic search algorithm expands paths in frontier, until it expands a goal
- Frontier is a stack $\longrightarrow$ depth-firt search
- Frontier is a queue $\longrightarrow$ breadth-firt search
- Frontier is a priority queue ordered by path cost $\longrightarrow$ least-cost-first search
- Frontier is a priority queue ordered by $f(p) = cost(p) + h(p)$ $\longrightarrow$ A^* search
- Cycle pruning prunes paths that loop back on themselves
- Multiple-path pruning prunes paths to nodes that have already been expanded.
- Depth-first branch-and-bound combines space saving of depth-first search with the optimality of A^*.

Clicker Question

Which of the following is false:

A Iterative deepening saves space over breadth-first search

B Iterative deepening finds the same answer as breadth-first search

C Iterative deepening runs faster than breadth-first search

D Iterative deepening recomputes elements of the frontier that breadth-first search stores

Clicker Question

What is not true of depth-first branch-and-bound

A It always find an optimal solution if the bound starts as a finite overestimate of the cost of an optimal solution

B The bound always reduces whenever a new best solution is found

C It does not necessarily halt if the bound is infinite

D It uses linear space even if used with multiple path pruning

E If it halts and returns a path, the path is an optimal solution even with loop detection

(Assume the graph and heuristic have the properties assumed in the proof that A^* is admissible.)

Clicker Question

What is true of depth-first branch-and-bound search

A It use as much space as A^*

B It always halts and finds an optimal solution (even with infinite initial bound)

C It can get confused if there are multiple optimal solutions

D The first time *best* is assigned a path (inside the loop), it has found a best solution

E None of the above

Assignment 2 Discussion

- Heuristics h1 and h2

Assignment 2 Discussion

- Heuristics h1 and h2
- Does the minus matter?

Assignment 2 Discussion

- Heuristics h1 and h2
- Does the minus matter? Why?

Assignment 2 Discussion

- Heuristics h1 and h2
- Does the minus matter? Why?
- When is depth-first better than breadth-first search?

Assignment 2 Discussion

- Heuristics h1 and h2
- Does the minus matter? Why?
- When is depth-first better than breadth-first search? When is breadth-first better than depth-first?

Assignment 2 Discussion

- Heuristics h1 and h2
- Does the minus matter? Why?
- When is depth-first better than breadth-first search?
  When is breadth-first better than depth-first?
  What is A^* better than breadth-first or depth-first searches?

Assignment 2 Discussion

- Heuristics h1 and h2
- Does the minus matter? Why?
- When is depth-first better than breadth-first search?
  When is breadth-first better than depth-first?
  What is A^* better than breadth-first or depth-first searches?
  When is A^* worse than others?

Review: So far...

- An agent acts in an environment
- Agent has access to: abilities, goals/preferences, prior knowledge, observations, past experiences
- Search is used to find paths in graphs
- Search algorithms differ in how elements of frontier are selected
- Multiple-path pruning and loop pruning can reduce search
- Depth-bounded depth-first search (as used in iterative deepening and branch-and-bound) can save space

Today: Learning Objectives

Today: Constraint Satisfaction Problems

At the end of the class you should be able to:

- recognize and represent constraint satisfaction problems

Posing a Constraint Satisfaction Problem

A CSP is characterized by

- A set of variables $V_1, V_2, \ldots, V_n$.
- Each variable V_i has an associated domain D_{V_i} which specifies the set of possible values the variable can take.
  (We assume domains are finite.)
- A possible world or total assignment is an assignment of a value to each variable.

Posing a Constraint Satisfaction Problem

A CSP is characterized by

- A set of variables $V_1, V_2, \ldots, V_n$.
- Each variable V_i has an associated domain D_{V_i} which specifies the set of possible values the variable can take.
  (We assume domains are finite.)
- A possible world or total assignment is an assignment of a value to each variable.
- A hard constraint on a subset of variables specifies which combination of values are legal
- A solution to CSP (a model) is possible world that satisfies all the constraints.

Simple Examples

Example 1:

- Variables: A, B, C
- Domains: $\{1, 2, 3, 4\}$
- Constraints $A < B$, $B < C$

Simple Examples

Example 1:

- Variables: A, B, C
- Domains: $\{1, 2, 3, 4\}$
- Constraints $A < B$, $B < C$

Example 2:

- Variables: A, B, C, D
- Domains: $\{1, 2, 3, 4\}$
- Constraints $A < B$, $B < C, C < D$

Simple Examples

Example 1:

- Variables: A, B, C
- Domains: $\{1, 2, 3, 4\}$
- Constraints $A < B$, $B < C$

Example 2:

- Variables: A, B, C, D
- Domains: $\{1, 2, 3, 4\}$
- Constraints $A < B$, $B < C, C < D$

Example 3:

- Variables: A, B, C, D, E
- Domains: $\{1, 2, 3, 4\}$
- Constraints $A < B$, $B < C, C < D, D < E$

CSP variants

- determine whether or not a model exists

CSP variants

- determine whether or not a model exists
- find a model

CSP variants

- determine whether or not a model exists
- find a model
- find all models

CSP variants

- determine whether or not a model exists
- find a model
- find all models
- count the number of models

CSP variants

- determine whether or not a model exists
- find a model
- find all models
- count the number of models
- find the best model given some model quality
  - soft constraints specify preferences

CSP variants

- determine whether or not a model exists
- find a model
- find all models
- count the number of models
- find the best model given some model quality
  - soft constraints specify preferences
- determine whether some property holds in all of the models

Example: scheduling activities

- Variables: A, B, C, D, E that represent the starting times of various activities.
- Domains: $\mathrm{D}_A = \{1, 2, 3, 4\}$, $\mathrm{D}_B = \{1, 2, 3, 4\}$, $\mathrm{D}_C = \{1, 2, 3, 4\}$, $\mathrm{D}_D = \{1, 2, 3, 4\}$, $\mathrm{D}_E = \{1, 2, 3, 4\}$

Example: scheduling activities

- Variables: A, B, C, D, E that represent the starting times of various activities.
- Domains: $\mathrm{D}_A = \{1, 2, 3, 4\}$, $\mathrm{D}_B = \{1, 2, 3, 4\}$, $\mathrm{D}_C = \{1, 2, 3, 4\}$, $\mathrm{D}_D = \{1, 2, 3, 4\}$, $\mathrm{D}_E = \{1, 2, 3, 4\}$
- What are some possible worlds?

Example: scheduling activities

- Variables: A, B, C, D, E that represent the starting times of various activities.
- Domains: $\mathrm{D}_A = \{1, 2, 3, 4\}$, $\mathrm{D}_B = \{1, 2, 3, 4\}$, $\mathrm{D}_C = \{1, 2, 3, 4\}$, $\mathrm{D}_D = \{1, 2, 3, 4\}$, $\mathrm{D}_E = \{1, 2, 3, 4\}$
- What are some possible worlds?
- How many possible worlds are there?

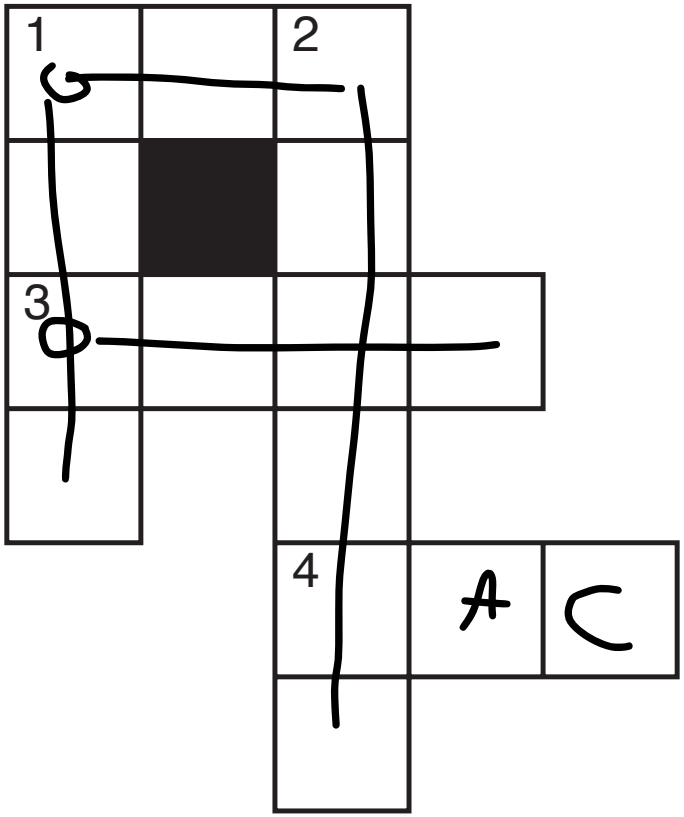
$4 \times 4 \times 4$

Example: scheduling activities

- Variables: A, B, C, D, E that represent the starting times of various activities.
- Domains: $\mathrm{D}_A = \{1, 2, 3, 4\}$, $\mathrm{D}_B = \{1, 2, 3, 4\}$, $\mathrm{D}_C = \{1, 2, 3, 4\}$, $\mathrm{D}_D = \{1, 2, 3, 4\}$, $\mathrm{D}_E = \{1, 2, 3, 4\}$
- What are some possible worlds?
- How many possible worlds are there?
- Constraints:

$$(B \neq 3) \wedge (C \neq 2) \wedge (A \neq B) \wedge (B \neq C) \wedge (C < D) \wedge (A = D) \wedge (E < A) \wedge (E < B) \wedge (E < C) \wedge (E < D) \wedge (B \neq D).$$

Example: Crossword Puzzle

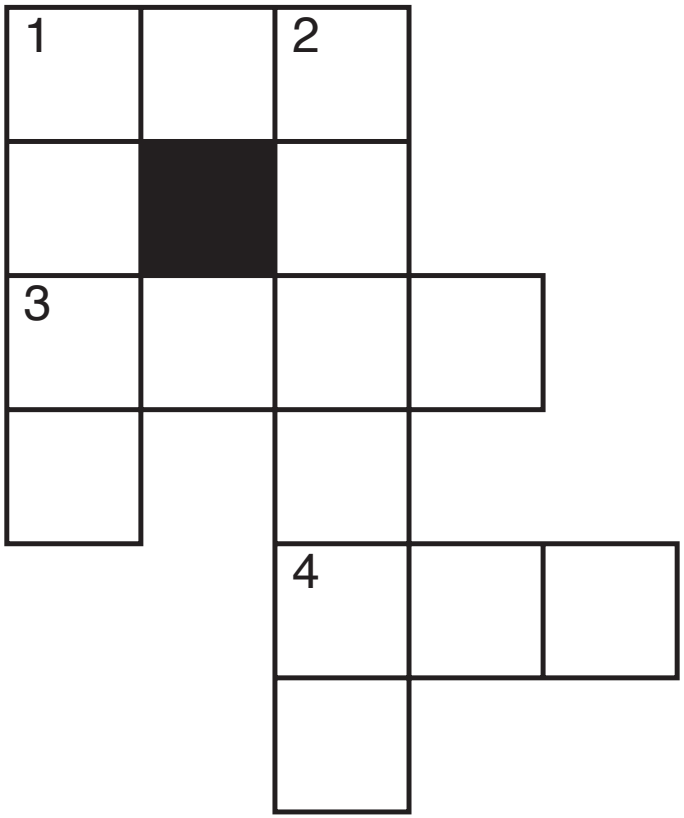


Words:

ant, big, bus, car, has
book, buys, hold,
lane, year
beast, ginger, search,
symbol, syntax

- What are the variables?
- What are their domains?

Example: Crossword Puzzle

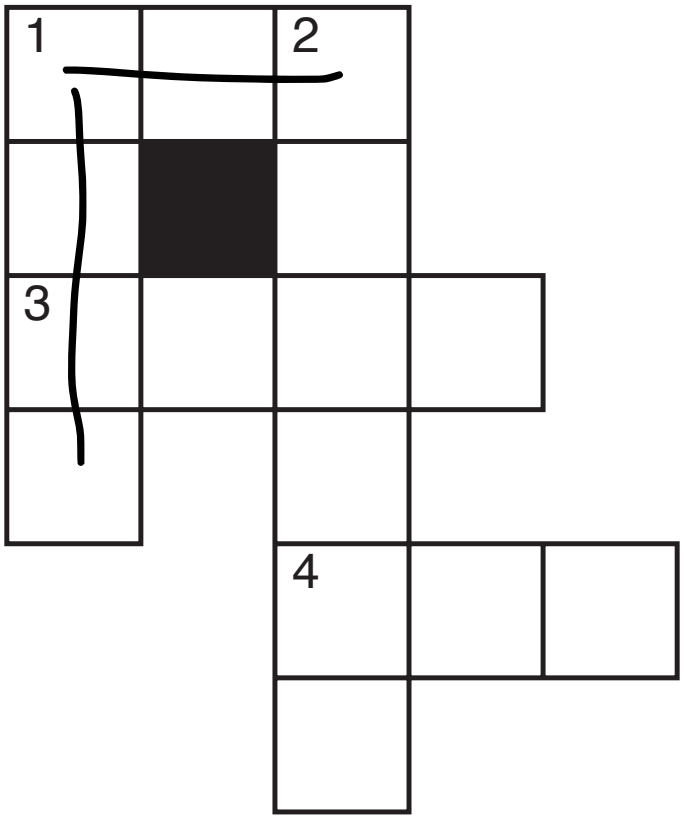


Words:

ant, big, bus, car, has
book, buys, hold,
lane, year
beast, ginger, search,
symbol, syntax

- What are the variables?
- What are their domains?
- How many possible worlds are there? 15^5

Example: Crossword Puzzle

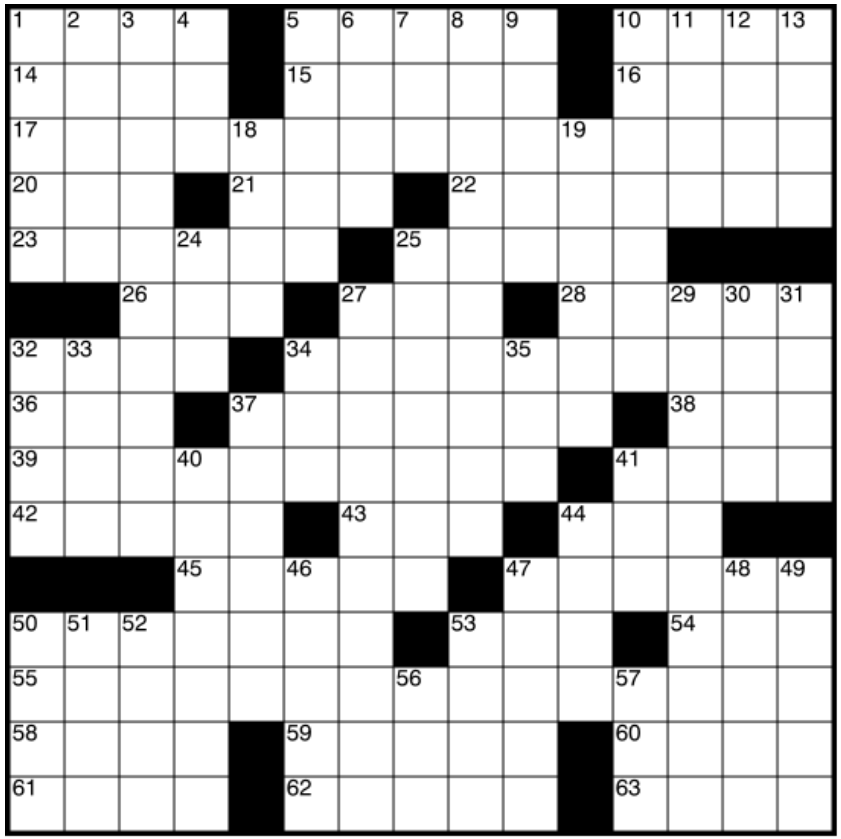


Words:

ant, big, bus, car, has
book, buys, hold,
lane, year
beast, ginger, search,
symbol, syntax

- What are the variables?
- What are their domains?
- How many possible worlds are there?
- What are the constraints?

Example: Crossword Puzzle



Suppose there are 10,000 words of each length (from 2 to 10) and 70 positions to put words

- How many possible worlds are there?

10^{70}

Clicker Question

Which of the following is not true of constraint satisfaction problems:

A People often solve constraint satisfaction problems for recreation

B A CSP is defined by variables, domains and constraints

C A constraint specifies whether an assignment of values to some of the variables is legal

D The domain of a variable specifies the values that the variable can take

E There is only ever one way to define a problem as a CSP

Clicker Question

Suppose there were 17 variables, each with domain size 100. How many possible worlds are there?

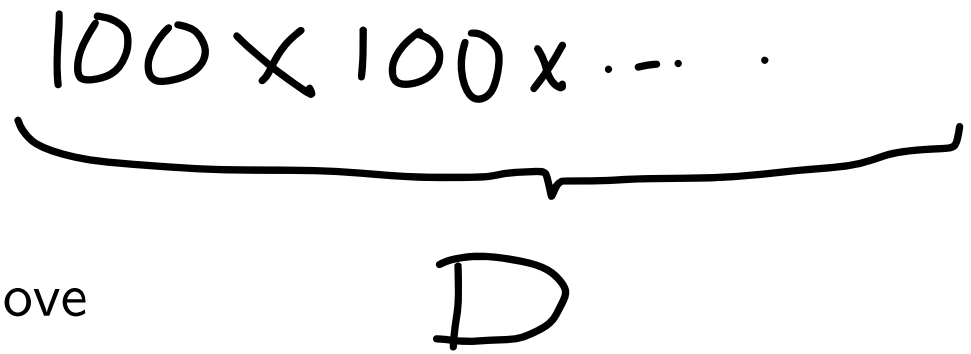
A 1700

B 117

C 17^{100}

D 100^{17}

E None of the above



Example: Sodoku

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

- What are the variables?

Example: Sodoku

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

- What are the variables?
- What is their domain?

Example: Sodoku

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

- What are the variables?
- What is their domain?
- How many possible worlds are there?

Example: Sodoku

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

- What are the variables?
- What is their domain?
- How many possible worlds are there?
- What are the constraints?

Scheduling final exams

UBC exam scheduling is done by an AI system:

- 13 exam days, 52 timeslots
- 30,000 students take exams
- 1,700 sections with exams
- 105,000 student-exam pairs
- 274 rooms across 38 buildings

Scheduling final exams

UBC exam scheduling is done by an AI system:

- 13 exam days, 52 timeslots
- 30,000 students take exams
- 1,700 sections with exams
- 105,000 student-exam pairs
- 274 rooms across 38 buildings

- What are the variables? sections

Scheduling final exams

UBC exam scheduling is done by an AI system:

- 13 exam days, 52 timeslots
- 30,000 students take exams
- 1,700 sections with exams
- 105,000 student-exam pairs
- 274 rooms across 38 buildings

- What are the variables?
- What are the domains? times

Scheduling final exams

UBC exam scheduling is done by an AI system:

- 13 exam days, 52 timeslots
- 30,000 students take exams
- 1,700 sections with exams
- 105,000 student-exam pairs
- 274 rooms across 38 buildings

- What are the variables?
- What are the domains?
- How many possible worlds are there?

52

Scheduling final exams

UBC exam scheduling is done by an AI system:

- 13 exam days, 52 timeslots
- 30,000 students take exams
- 1,700 sections with exams
- 105,000 student-exam pairs
- 274 rooms across 38 buildings

- What are the variables?
- What are the domains?
- How many possible worlds are there?
- What are the constraints?

UBC Exam Scheduling Hard Constraints

- There can't be more than 30 conflicts for a section
- Allowable times for each exam
- Allowable rooms for each exam
- Requested room features for each exam
- Unrelated exams cannot share a room
- Cross-listed courses must have the same exam time
- Evening courses must have evening exams

UBC Exam Scheduling Soft Constraints

Try to minimize:

- Conflicts
- Students with 2+ exams on the same day
- Students with 3+ exams in 4 consecutive timeslots
- Students with back-to-back exams
- Students with less than 8 timeslots between exams
- Preferred times for each exam
- Preferred rooms for each exam
- Room capacities
- First-year exams on the last two days (Fall exams)
- Fourth-year exams on the last two days (Spring exams)