UCLA Samueli School of Engineering

CS161 WEEK4 DISCUSSION 1C

Danfeng Guo

Contributed by Yewen W’ and Shirley C’s previous course materials

Agenda

CSP

TWO-PLAYER

Propositional logic

CSP

FORMULATION

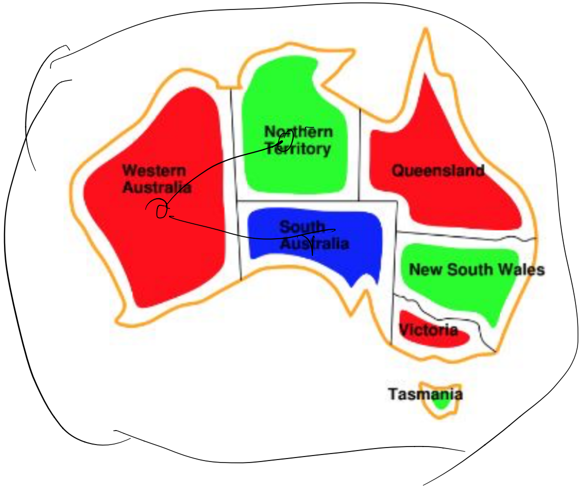
Variables

Domains

Constraints

State

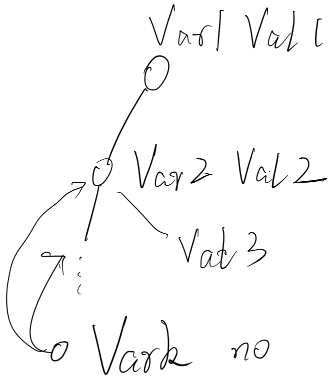
- Complete assignment
- Partial assignment
- Consistent assignment



CSP

BACKTRACKING DFS

- Choose a var and assign a val
- Backtrack when no legal assignments
- Search until it fails



CSP

BACKTRACKING DFS

- How to select a variable?
  - the one with the most constraints / least-remaining values
  - least branches
- How to assign a value?
  - least constraining values

CSP

ARC CONSISTENCY

X_i is arc-consistent with respect to another variable X_j if for every value in the current domain D_i there is some value in the domain D_j that satisfies the binary constraint on the arc $(X_i\ X_j)$

Domain Refine

Given a set of all arcs

Each time pop an arc and check its domain

- If unchanged, move to the next one
- If shrinks, update
- If empty, fail

Stop until all arcs are unchanged

eg: Domain: 0~9

Constraint: $Y = X^2$

$X = \{0, 1, 2, 3\}$

$Y = \{0, 1, 4, 9\}$

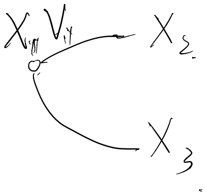
CSP

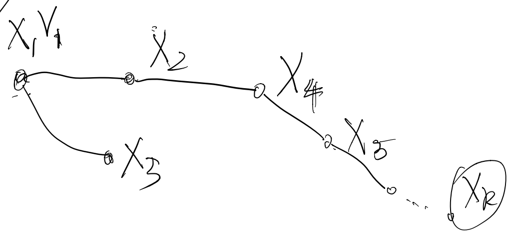
Forward Checking

- Check remaining values for connected unassigned variables

Maintaining Arc Consistency (MAC)

- Pick a neighbor of Var X, call it Var Y
- If domain of Y reduces, do MAC on Y
- Keep checking if all are unchanged.





CSP

PRACTICE

The domain for each variable is {1, 2, 3, 4}
Constraints:

1. a cannot be 3 or 4
2. b cannot be 4
3. Connected variables cannot be the same

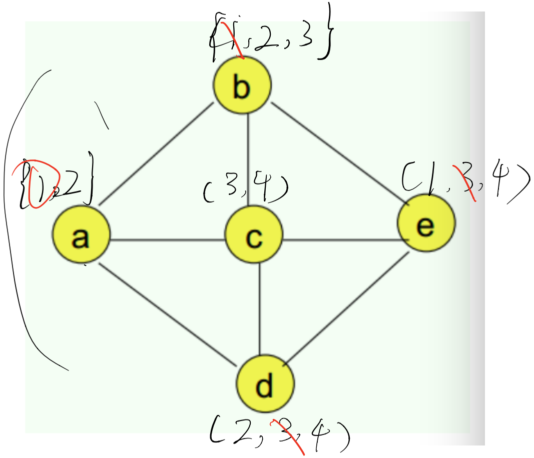
Start? a:1

b:2

c:3

tie d:2 | d:1

e:1 e:4



TWO-PLAYER

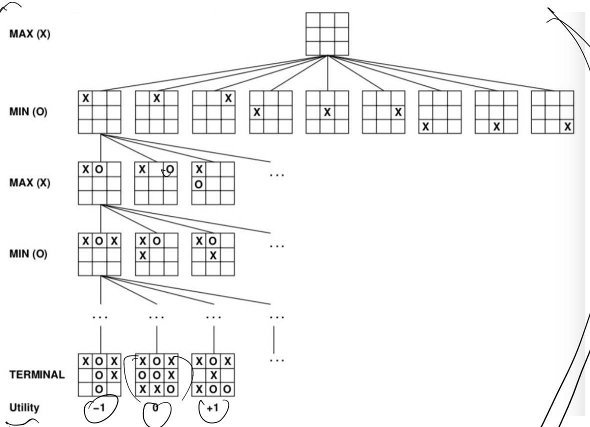
BASICS

- Perfect/Imperfect
- Deterministic/Chance

FORMULATION AS SEARCH PROBLEM

- Initial state, Player, Action, Result, Goal test, Utility

evaluate



TWO-PLAYER

MINIMAX ALGORITHM

Assume a deterministic & perfect game

Choose the move with highest achievable playoff against the best choice of the other

TWO-PLAYER

MINMAX ALGORITHM

Complete: Y

Optimal: Y

Time: O(b^m)

Space: O(b^m)

TWO-PLAYER

ALPHA-BETA PRUNING

No need to explore every path

Alpha: maximum lower bound of possible solutions

Beta: minimum upper bound of possible solutions

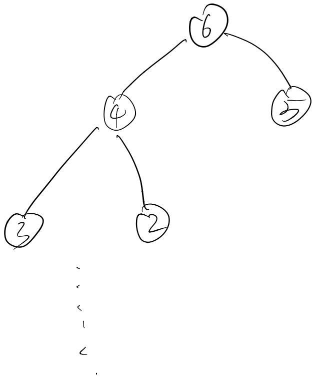
For min: if children's alpha >= parent's beta, prune

For max: if children's beta <= parent's alpha, prune

TWO-PLAYER

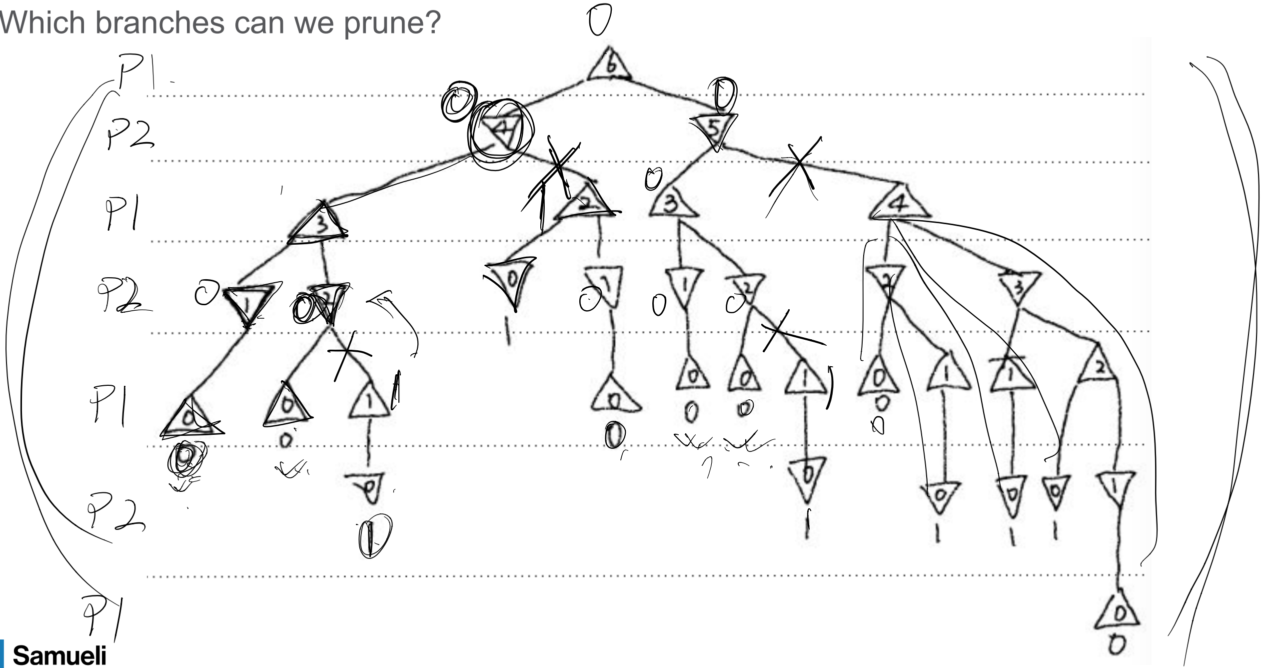
EXAMPLE

- Start with 6 stones
- Each time, one can take either one or two
- The one who takes the last one wins



TWO-PLAYER

Which branches can we prune?



Propositional Logic

$a \leftrightarrow b : \begin{cases} a \to b \\ b \to a \end{cases}$

Syntax: "and" "or" $a \to b$

- ¬(not), ∧ (conjunction), ∨ (disjunction), → (implication), ↔(biconditional)

Semantics:

- ¬ f - True iff f is false
- f ∨ g - True iff at least one of f or g is True
- f ∧ g - True iff both f and g are True
- f → g - False iff f is true and g is false
- f ↔ g - True iff both f and g has same value

$f \to g = \neg(f \wedge \neg g) = \neg f \vee g$

Propositional Logic

Validity:

- A sentence is valid if its true in all models

Satisfiability:

- A sentence is satisfiable if it is true in some models
- A sentence is unsatisfiable if it is true in no models

X_1	X_2	X_3	Y
T	T	F	1
F	F	F	0
⋮			

Propositional Logic

a	b	$a \lor b$	a	b	$a \land b$
0	0	0	0	0	0
0	1	1	0	1	0
1	0	1	1	0	0
1	1	1	1	1	1

CNF
- Conjunction of disjunctions $(a \lor b) \land (b \lor c)$

DNF
- Disjunction of conjuctions $(a \land b) \lor (b \land c)$

HORN
- Horn clauses: at most one positive. Conjunction of horn clauses.

NNF
- Negation occurs only directly in front of atoms

Propositional Logic

CONVERSION USING LOGIC EQUIVALENCE

$(\alpha \land \beta)$	$\equiv$	$(\beta \land \alpha)$	commutativity of $\land$
$(\alpha \lor \beta)$	$\equiv$	$(\beta \lor \alpha)$	commutativity of $\lor$
$((\alpha \land \beta) \land \gamma)$	$\equiv$	$(\alpha \land (\beta \land \gamma))$	associativity of $\land$
$((\alpha \lor \beta) \lor \gamma)$	$\equiv$	$(\alpha \lor (\beta \lor \gamma))$	associativity of $\lor$
$\neg(\neg\alpha)$	$\equiv$	α	double-negation elimination
$(\alpha \Rightarrow \beta)$	$\equiv$	$(\neg\beta \Rightarrow \neg\alpha)$	contraposition
$(\alpha \Rightarrow \beta)$	$\equiv$	$(\neg\alpha \lor \beta)$	implication elimination
$(\alpha \Leftrightarrow \beta)$	$\equiv$	$((\alpha \Rightarrow \beta) \land (\beta \Rightarrow \alpha))$	biconditional elimination
$\neg(\alpha \land \beta)$	$\equiv$	$(\neg\alpha \lor \neg\beta)$	De Morgan
$\neg(\alpha \lor \beta)$	$\equiv$	$(\neg\alpha \land \neg\beta)$	De Morgan
$(\alpha \land (\beta \lor \gamma))$	$\equiv$	$((\alpha \land \beta) \lor (\alpha \land \gamma))$	distributivity of $\land$ over $\lor$
$(\alpha \lor (\beta \land \gamma))$	$\equiv$	$((\alpha \lor \beta) \land (\alpha \lor \gamma))$	distributivity of $\lor$ over $\land$

Qs: $A \Leftrightarrow (B \lor C)$

Propositional Logic

$A \Leftrightarrow (B \vee C)$

① Remove $\Leftrightarrow$

$(A \Rightarrow (B \vee C)) \wedge ((B \vee C) \Rightarrow A)$

② Remove $\Rightarrow$

$(\neg A \vee B \vee C) \wedge (\neg(B \vee C) \vee A)$

③ $\neg$

$(\neg A \vee B \vee C) \wedge ((\neg B \wedge \neg C) \vee A)$

④ $(\neg A \vee B \vee C) \wedge (\neg B \vee A) \wedge (\neg C \vee A)$

HOMEWORK3

NEXT-STATE

Consider the following situations

- Blank/Star in the front
- Box/BoxStar in the front → corner
  - Can push
  - Cannot push
- Wall in the front
- Deadlock check

HOMEWORK3

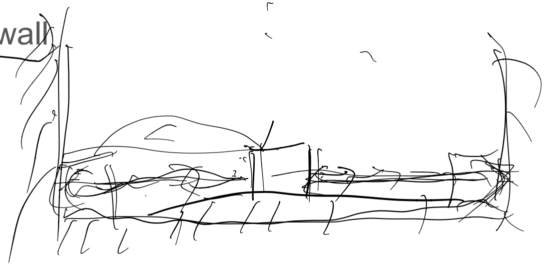
HEURISTICS

Manhattan Distance would be fine

- Dist from keeper to box + Dist from box to goal

The key to accelerate is to avoid ‘deadlock’ states as early as possible.

- Some states are ‘dead’ but we need to search a long way to find that.
  - A box on the corner
  - A box reaches the wall but no stars along the wall
  - Two boxes sit together along the wall
  - etc



HOMEWORK4

DESIGN A SAT SOLVER

- Given a propositional sentence in CNF, return whether it is satisfiable
- Take it as CSP.
- Represent CNF with list
  - Variable is from 1 to n
  - Clause is a list of integers
  - Negative sign means negation
- Return one possible solution

$(A \vee B) \wedge (C \vee D) \ldots$

T F

(1 2 3)

(-1 2 3)

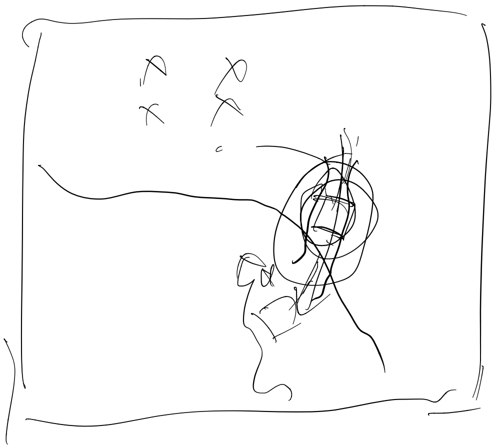
$(X \vee \neg Y \vee Z) \wedge \neg X \wedge (\neg Y \vee \neg Z)$

((1 -2 3) (-1) (-2 -3))

HOMEWORK4

BREAK INTO PARTS

- Goal-test:
  - Check whether a clause is valid
  - Check whether a sentence is valid
- Backtracking DFS
  - Select a variable (or not select)
  - Assign a value to a variable



Q&A