Game Computational Complexity Basics

Kyle Burke

Florida Southern College

Games at Mumbai, January 23, 2024

Acronym

Games At Mumbai (GAM _ _)

- Evenly Symmetric?
- w/Easy Steps?
- w/Easy Rules?
- Extra Spicy?

Talk Plan

- ACGT: Algorithmic Combinatorial Game Theory
- Describe PSPACE and QSAT
- Reductions and Complexity
- Show reduction: QSAT $\rightarrow$ GEOGRAPHY
- Computational Complexity of COL and NOGO.
- Broader landscape of games.

Note: interruptible!

ACGT

- Big Question (outcome classes): Who is winning? Who is the winner?
- How hard is it to determine the winner?
- How hard is it to compute the winner?
- How long does an algorithm run to compute the winner?
- How long does the *best* algorithm run to compute the winner?
- How long does the *best* algorithm run to compute the winner in the worst cases?
- How does that change as the size of the game position grows?
- We want to be thinking about the last version of this question.

QSAT

- Let's start off with a game! BOOLEAN FORMULA:
  - Players: True vs. False
  - Position:
    - List of variables with assignments: (True, False, Unassigned)
    - Formula in Conjunctive Normal Form (CNF).
  - Turn: Current player picks value for the next unassigned variable: True or False. Can't move if the formula is already obviously won by other player, or if the move would cause such an assignment.
  - Normal Play: If you can't make a move, you lose.
- Let's try playing: `http://kyleburke.info/DB/combGames/booleanFormula.html`

QSAT

- True wins going first exactly when the QSAT problem is true
- QSAT:

$$\exists x_0 : \forall x_1 : \exists x_2 : \forall x_3 : \exists x_4 : \forall x_5 :$$

$$(x_0 \vee \overline{x_1} \vee x_2) \wedge (\overline{x_2} \vee x_3 \vee x_4) \wedge (\overline{x_0} \vee x_2 \vee \overline{x_3}) \wedge (x_3 \vee \overline{x_4} \vee \overline{x_5})$$

- QSAT is canonical PSPACE-complete problem.

QSAT

- PSPACE: the class of all (T/F) computational problems that can be solved with a polynomial-amount of read/write space.
- Note: no restriction on the amount of time.
- Every problem in PSPACE can be reduced to QSAT in polynomial-time.
- $\forall X \in \mathrm{PSPACE} : \exists f : X \to QSAT$ where:
  - $\forall x \in X : x$ is a true instance of $X \Leftrightarrow f(x)$ is a true instance of $QSAT$.
  - f runs efficiently
- These transformations, *reductions*, are important in determining what problems can be solved.

QSAT

- What can we say about two problems, X and Y where there's a reduction $f : X \to Y$?
- What happens if there's an algorithm, B, that solves problem Y in polynomial time?
- Then there's an algorithm that solves X in polynomial time:
- $A(x)$:

```
y := f(x)
return B(y)
```

- Polynomial + polynomial = polynomial
- By contrapositive, if there's no efficient algorithm for X, then there's no efficient algorithm for Y.
- "Hardness follows a reduction."
  "Easiness salmons a reduction."

QSAT

- There are reductions from everything in PSPACE to QSAT.
- PSPACE-*complete*: QSAT is among hardest problems in PSPACE. (Best known algorithms are exponential. "Intractable".)
- Boolean Formula is also PSPACE-complete.
- That's good for games!
- Players use approximation, randomness, and heuristics instead of certainty.
- CGT: hardness/completeness results are positive instead of negative.

Geography

- Geography is another PSPACE-complete game!
- Played on a directed graph with a token on one vertex.
- Turn: move the token along an arc; can't ever return to the vertex left behind.
- Impartial Game: both players always have same options.
- Normal Play: Lose if there's no outgoing edges.
- Let's Play!
  `http://kyleburke.info/DB/combGames/geography.html`

GEOGRAPHY

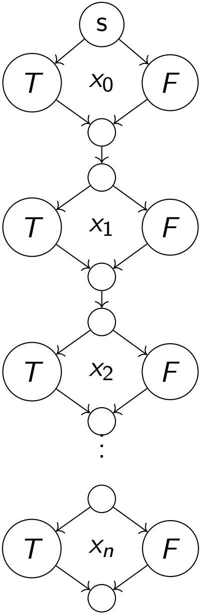
- Why is GEOGRAPHY PSPACE-complete?
- There's a reduction!
  f : BOOLEAN FORMULA $\rightarrow$ GEOGRAPHY [Schaefer 78[1]]
- (Very common first PSPACE-reduction.)
- Two steps: variable choosing, then evaluation.
- Note: could have BOOLEAN FORMULA choose all variables, then evaluate.

[1] "On the complexity of some two-person perfect-information games", T Schaefer, 1978

Geography

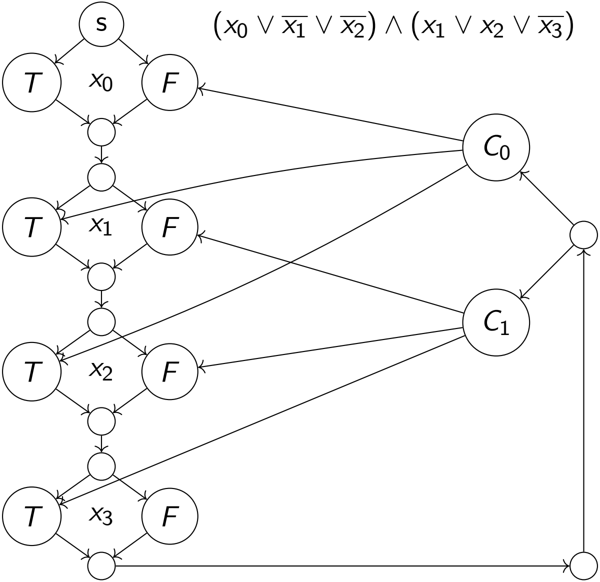
- Reduction: need to find a working function f.
- Reductions usually constructive, so we're building Geography instances.
- Needs to work for any Boolean Formula.
- *Not onto*: doesn't need to create every Geography position.

Geography



Geography

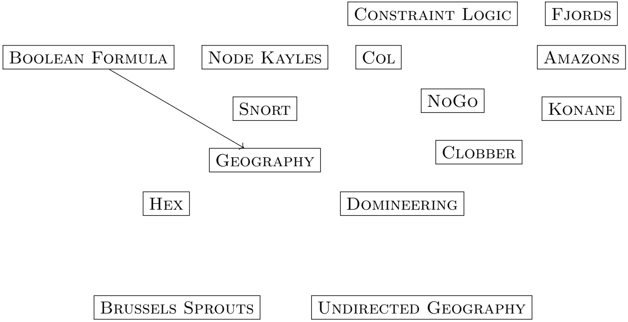
$(x_0 \vee \overline{x_1} \vee \overline{x_2}) \wedge (x_1 \vee x_2 \vee \overline{x_3})$



GEOGRAPHY

- $\forall G \in$ BOOLEAN FORMULA : True wins going first on $G \Leftrightarrow$ First player wins on $f(G)$
- Computational Hardness follows a reduction
- BOOLEAN FORMULA is PSPACE-complete $\Rightarrow$ GEOGRAPHY is PSPACE-complete.
- Now we can reduce from GEOGRAPHY.
- What about other rulesets?

Rulesets in PSPACE

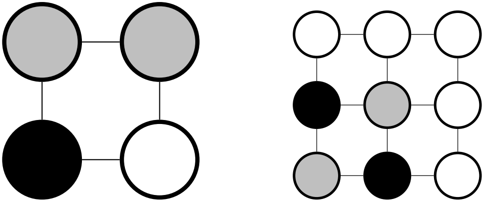


Col

- Let's do another one: COL:
  - Players: Blue vs. Red
  - Position: Undirected graph, with Red, Blue, and unpainted vertices.
  - Turn: Current player chooses an uncolored vertex and paints it their color. You're not allowed to pick a vertex adjacent to one already painted your color.
  - Normal Play: If you can't make a move, you lose.
  - Created in 1976 (or earlier) by Colin Vout. (Mentioned in On Numbers and Games)
- Let's try playing on a grid: `http://kyleburke.info/DB/combGames/col.html`

NoGo

- NoGo:
  - Players: Black vs. White.
  - Position: Undirected graph, with Black, White, and unpainted vertices.
  - Turn: Current player chooses an uncolored vertex and paints it their color. You're not allowed to paint a vertex that creates a connected component of either color that isn't adjacent to an empty vertex.



  - Normal Play: If you can't make a move, you lose.
  - Created in 2005 by John Moore. (Called "Anti-Atari Go".)
- Let's try playing on a grid: http://kyleburke.info/DB/combGames/noGo.html

Reduction: Col to NoGo

- Let's relate these with a reduction!
- f : Col → NoGo. [BH 2019[2]]
- Blue wins (going first) on Col position c

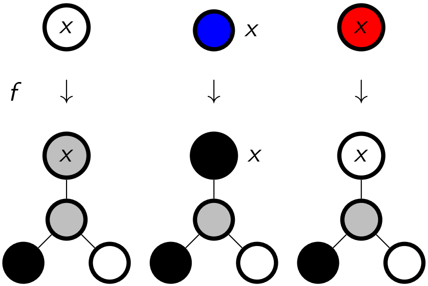
  $\Updownarrow$ (if and only if, or "exactly when")

  Black wins (going first) on NoGo position $f(c)$.
- "Local" reduction...

[2] "PSPACE-complete two-color planar placement games", KB, R Hearn, 2019.

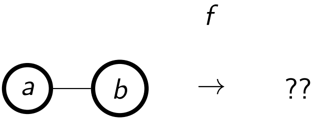
Reduction: Col to NoGo

- Can describe this locally: vertices and edges.
  - Vertices:

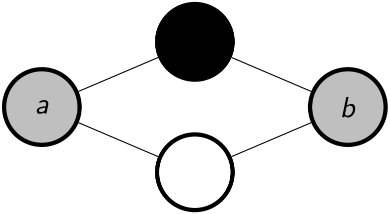


Reduction: Col to NoGo

- Edges:

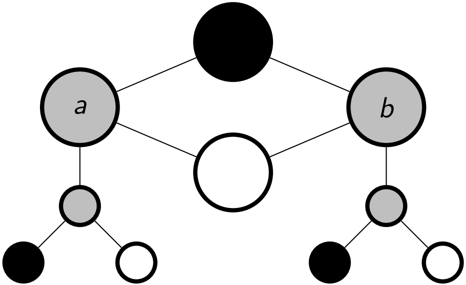


- What can we add so a and b can't both be white? What can we add so they can't both be black?



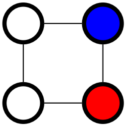
Reduction: Col to NoGo

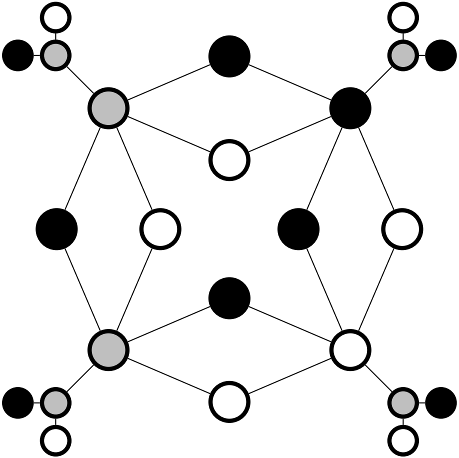
- Put them together:



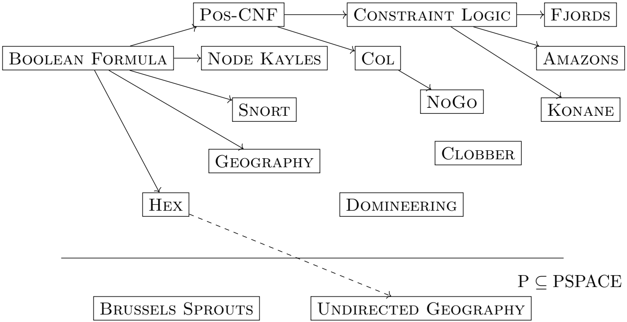
Reduction: Col to NoGo

- Example on a graph.





Reduction: Col to NoGo



Many: “On the complexity of some two-person perfect-information games”, T Schaefer, 1978.

Hex: “Hex ist PSPACE-vollständig.” S. Reisch, 1981.

Others: “Games, Puzzles, and Computation” R. Hearn, E. Demaine, 2009.

And: “PSPACE-complete two-color planar placement games.”, KB, R. Hearn, 2019.

Closing

- PSPACE-completeness (or harder) is good
- Find a reduction from PSPACE-complete ruleset to target game. (Not the other way around!)
- Fun! Building game boards.
- Different levels of reductions. Easy: play anywhere. More complicated: Play spaces in order.
- Common paper pattern: here's a new ruleset that's interesting... and here's the complexity.
- Unknown: CHOMP, SPROUTS, CLOBBER, DOMINEERING, GORGONS (Sprouts 2024[3] game)

Also, please be aware of FUN 2024[4]: June 4-8. Submission deadline: Feb 20.

Thank you!

[3] http://kyleburke.info/sprouts/sprouts2024/

[4] https://sites.google.com/unipi.it/fun2024