CGT class notes 2026

this is a brief lecture overview

I suggest that you also take your own notes

week 1 (lectures 1-2, Jan 6-8 2026)

- **syllabus** https://webdocs.cs.ualberta.ca/~hayward/cgt/jem/cgt.html
- **def'n: com'l game, normal play**
- **rules of clobber**
- **who wins alternating-full-grid 4×4 clobber?**
- **def'n: CG tree notation, CG {|} notation**
- **def'n: games 0, ∗, ↑, ↓**
- **rules of go** https://webdocs.cs.ualberta.ca/~hayward/355/rules.pdf

clobber

L player Left (also bLack, x)
R player Right (also whiTe, o)

N-psn (first player win)
P-psn (first player loss)
L-psn (L wins as 1st and 2nd player)
R-psn (R wins as 1st and 2nd player)

e.g. clobber

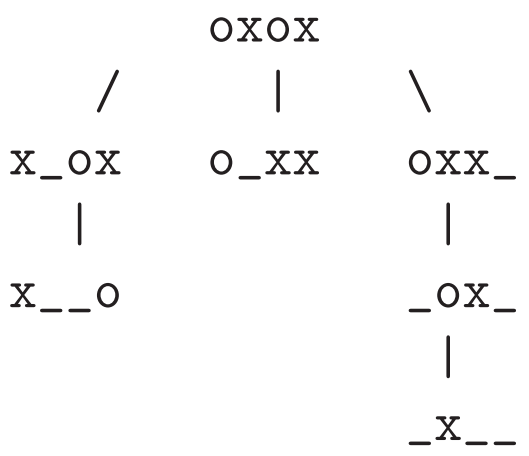
ox   in N

oxox ?

assume w.l.o.g. L plays first

tree of all possible continuations of the game ?

say x plays first



comb'l game: 2-player, alt.-turn, finite, perf.-info, deterministic

WLD game: comb'l game with every outcome win or loss or draw

win-strat: a guaranteed winning strategy

draw-strat: in WLD game, a guaranteed non-losing strategy

ptm: player-to-move *optm:* opponent-to-move

Zermelo's theorem: in WLDg, one of the following holds

- ptm has win-strat
- optm has win-strat
- ptm and optm each have draw-strat

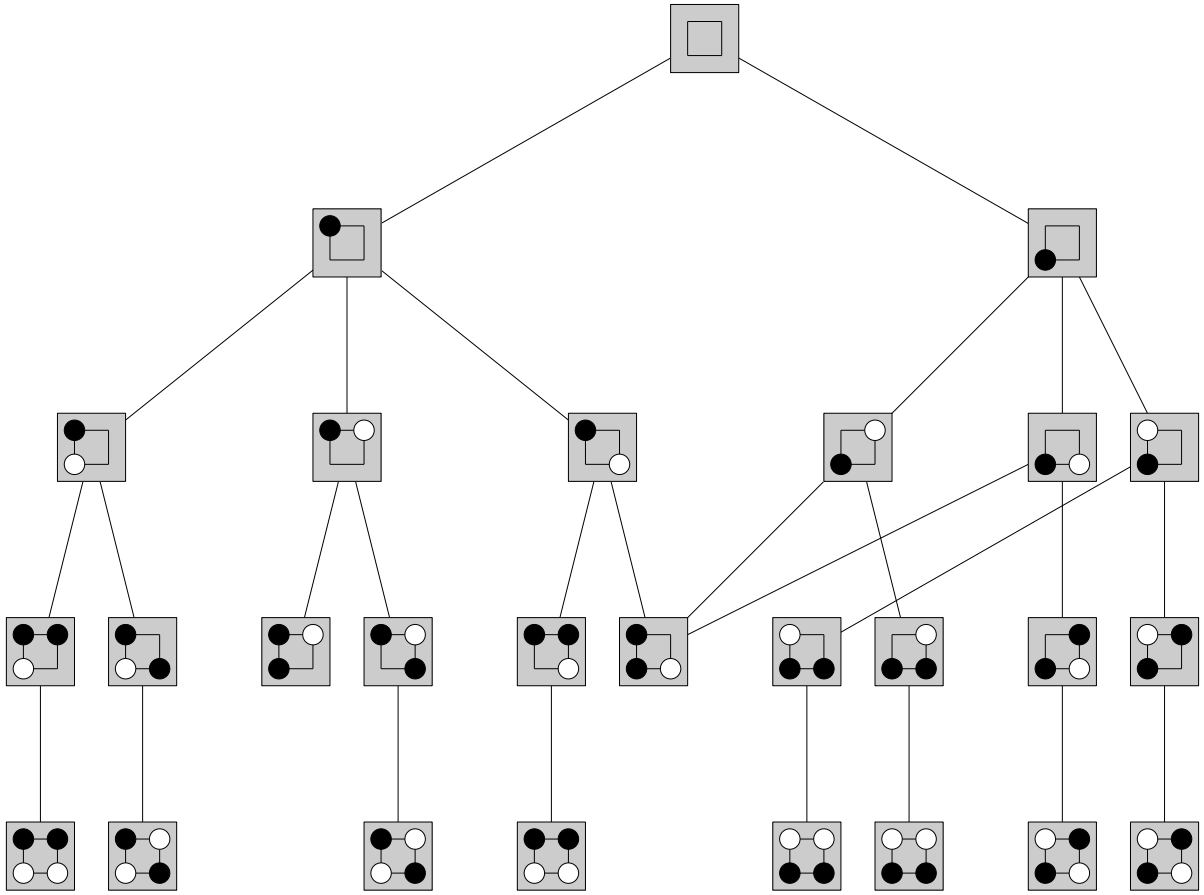
a simple game:

2p alt-turn

B wins with vertical column

W wins with horizontal row

next page: game dag (directed acyclic graph)



Z's theorem: proof by induction (sketch)

- **game g with options to $g_1, \ldots, g_t$**
- **by inductive assumption, theorem holds for $g_1, \ldots, g_t$, so each g_j outcome is $+$ (ptm-win), $-$ (optm-win), or 0 (ptm-draw and optm-draw)**
- **case A: some g_j outcome $-$, then g ptm-win (why?)**
- **case B: not case A and some g_j outcome 0, then g ptm-draw (why?) and optm-draw (why?)**
- **case C: not case A,B (each g_j outcome 1), then g optm-win (why?)**

exercise: where does prove use that g is finite?

back to linear clobber

- **who wins** `ox12` **?** `(oxoxoxoxoxox)`
- **game** `x { | }` **call this game 0**
- **game** `ox { 0 | 0 }` **call this game** $*$
- **game** `oxx { 0 | * }` **call this game up**
- **game** `oox { * | 0 }` **call this game down**
- **game** `oxox { 0, *, up | 0, *, down }`