Lecture 27

Beyond NP

Many complexity classes are worse, e.g. time 2^{2^n}, $2^{2^{2^n}}$, …

Others seem to be "worse" in a different sense, e.g., not in NP, but still exponential time. E.g., let

Lp = "assignment y satisfies formula x", ∈ P

Then :

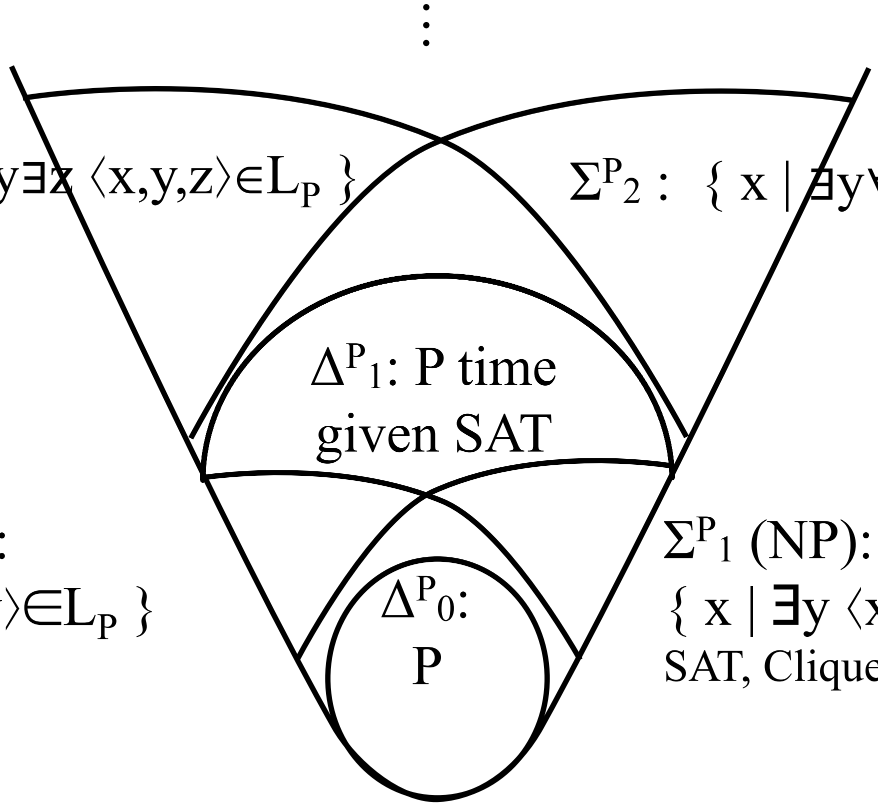
$\text{SAT} = \{ x \mid \exists y \ \langle x,y \rangle \in L_P \}$

$\text{UNSAT} = \{ x \mid \forall y \ \langle x,y \rangle \notin L_P \}$

$\text{QBF}_k = \{ x \mid \exists y_1 \forall y_2 \exists y_3 \ldots \partial_k \ y_k \ \langle x,y_1 \ldots y_k \rangle \in L_P \}$

$\text{QBF}_\infty = \{ x \mid \exists y_1 \forall y_2 \exists y_3 \ldots \qquad \langle x,y_1 \ldots \quad \rangle \in L_P \}$

The “Polynomial Hierarchy”



⋮

Π^P_2: $\{ x \mid \forall y \exists z \ \langle x,y,z \rangle \in L_P \}$

Σ^P_2: $\{ x \mid \exists y \forall z \ \langle x,y,z \rangle \in L_P \}$

Δ^P_1: P time given SAT

Π^P_1 (co-NP):
$\{ x \mid \forall y \ \langle x,y \rangle \in L_P \}$
UNSAT,…

Δ^P_0:
P

Σ^P_1 (NP):
$\{ x \mid \exists y \ \langle x,y \rangle \in L_P \}$
SAT, Clique, VC, HC, Knap,…

Potential Utility: It is often easy to give such a quantifier-based characterization of a language; doing so suggests (but doesn’t prove) whether it is in P, NP, etc. and suggests candidates for reducing to it.

Examples

QBF_k in Σ_k^P

Given graph G, integers j & k, is there a set U of ≤ j vertices in G such that every k-clique contains a vertex in U?

Given graph G, integers j & k, is there a set U of ≥ j vertices in G such removal of any k edges leaves a Hamilton path in U?

Space Complexity

DTM M has space complexity S(n) if it halts on all inputs, and never visits more than S(n) tape cells on any input of length n.

NTM …on any input of length n on any computation path.

DSPACE(S(n)) = { L | L acc by some DTM in space O(S(n)) }

NSPACE(S(n)) = { L | L acc by some NTM in space O(S(n)) }

Model-independence

As with Time complexity, model doesn't matter much. E.g.:

SPACE(n) on DTM ≈ O(n) bytes on your laptop

Why? Simulate each by the other.

Space vs Time

Time T $\subseteq$ Space T

Pf: no time to use more space

Space T $\subseteq$ Time 2^{cT}

Pf: if run longer, looping

Space seems more powerful

Intuitively, space is reusable, time isn't

Ex.: SAT $\in$ DSPACE(n)

Pf: try all possible assignments, one after the other

Even more:

$\mathrm{QBF}_k = \{ \exists y_1 \forall y_2 \exists y_3 \ldots \partial_k\, y_k\, x \mid \langle x, y_1 \ldots y_k \rangle \in L_P \} \in \mathrm{DSPACE}(n)$

$\mathrm{QBF}_\infty = \{ \exists y_1 \forall y_2 \exists y_3 \ldots \quad x \mid \langle x, y_1 \ldots \rangle \in L_P \} \in \mathrm{DSPACE}(n)$

PSPACE = Space($n^{O(1)}$)

NP $\subseteq$ PSPACE

pf: depth-first search of NTM computation tree

Games

2 player "board" games

E.g., checkers, chess, tic-tac-toe, nim, go, …

A finite, discrete "game board"

Some pieces placed and/or moved on it

"Perfect information": no hidden data, no randomness

Player I/Player II alternate turns

Defined win/lose configurations (3-in-a-row; checkmate; …)

Winning strategy:

∃move by player I ∀moves by II ∃ a move by I ∀… I wins.

Game Tree

Config:

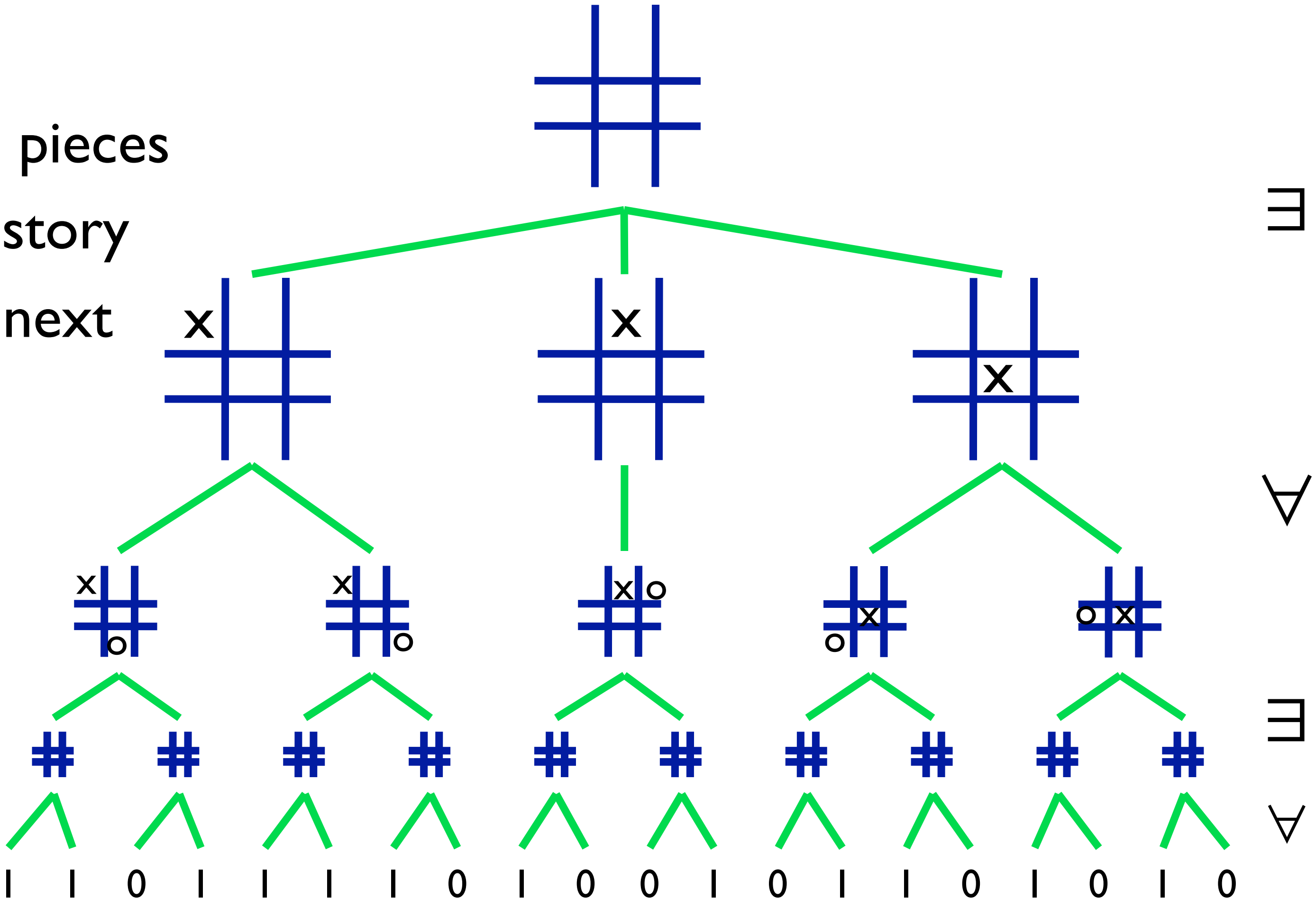
Where are pieces

Relevant history

Who goes next

Play:

All moves



Win/lose:

Game Tree

Config:

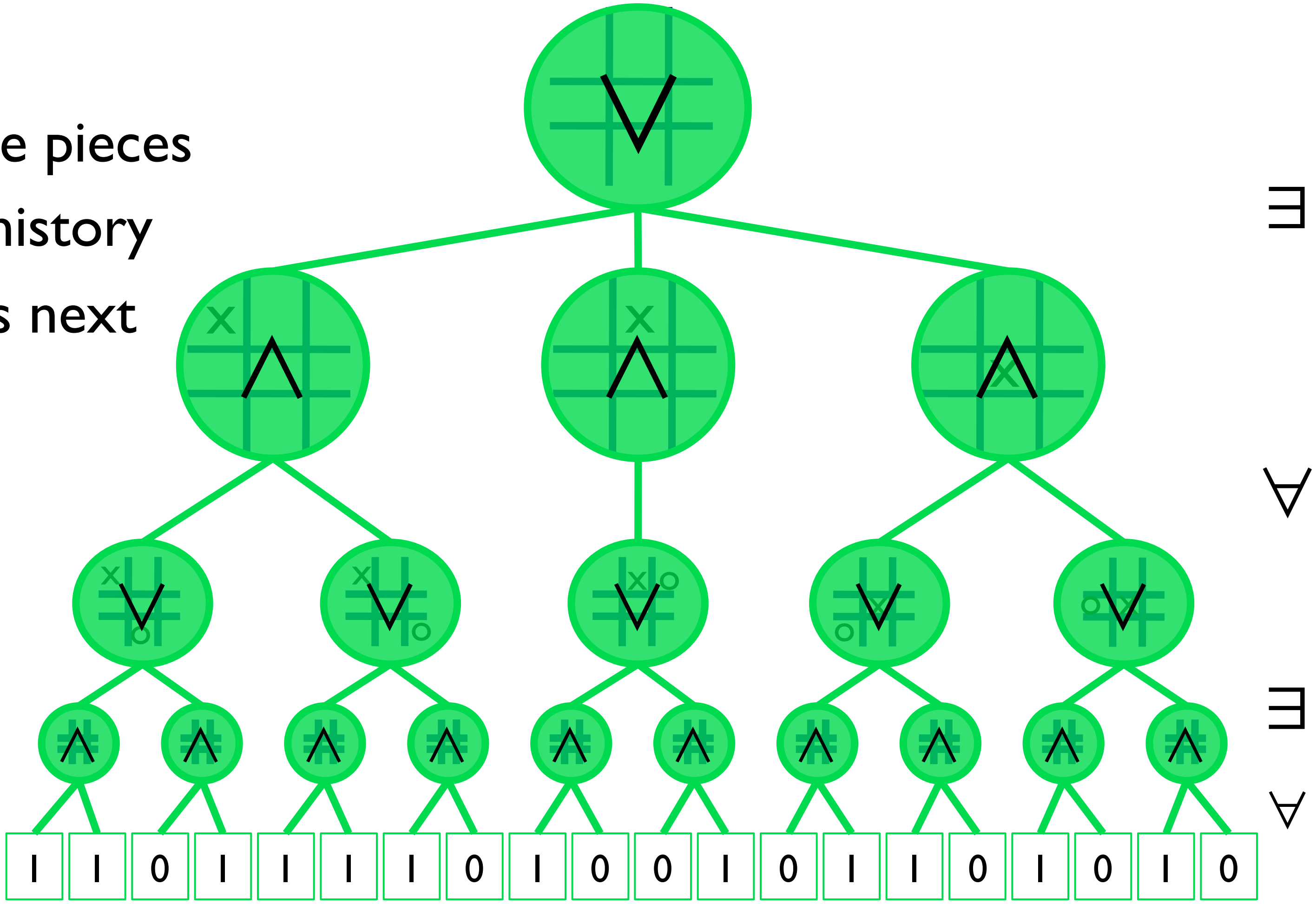
Where are pieces

Relevant history

Who goes next

Play:

All moves



Win/lose:

Winning Strategy

Config:

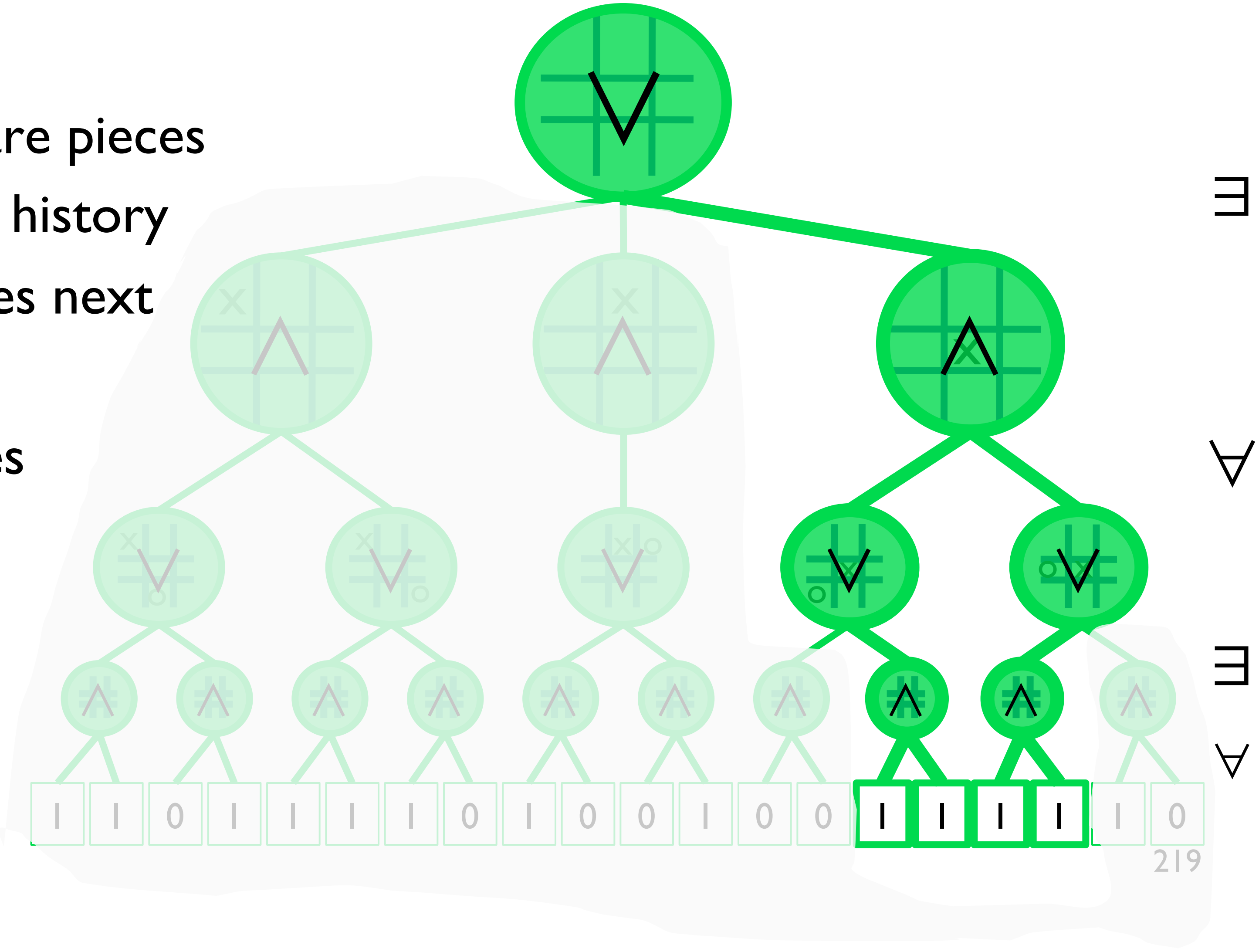
Where are pieces

Relevant history

Who goes next

Play:

All moves



Win/lose:

Complexity of 2 person, perfect information games

From above, *IF*

config (incl. history, etc.) is poly size

only poly many successors of one config

each computable in poly time

win/lose configs recognizable in poly time, and

game lasts poly # moves

THEN

in PSPACE!

Pf: depth-first search of tree, calc node values as you go.