Game: Hedging Edges
- The board is a planar graph
- The move is labeling an edge with a unit square label (one label per edge)
- Examples:

- Impartial game
  - Every position has a value

\[
\equiv \{\_\} \equiv \{*0\} \equiv *1
\]
\[
\begin{align*}
\text{\textbullet} & \equiv \{ \text{\textbullet} \} \\
\text{\textbullet} & \equiv \{ \text{\textbullet} , \text{\textbullet} , \text{\textbullet} , \text{\textbullet} , \text{\textbullet} , \text{\textbullet} \} \\
\text{\textbullet} & \equiv \{ \text{\textbullet} , \text{\textbullet} , \text{\textbullet} \} \equiv *1 \\
\text{\textbullet} & \equiv \{ \} \equiv \{ *0 \} \equiv *1
\end{align*}
\]
Complexity
- Question:
  - Given a position in the Hedging Edges game, is it possible to label all the remaining edges?

Reduction: reduce form planar 3-SAT
- Variable Gadget

- Propagation gadget: move info from one part to the other
- Turning Gadget

- Clause Gadget

Figure 2: Encoding the clause of a Boolean formula; here \( \overline{x} \lor y \lor \overline{z} \)
Moving to QBF

What do we know?
$P \subseteq NP \subseteq \text{PSPACE} \subseteq \text{NEXPTIME} \subseteq \text{EXPSPACE}$
$NPSPACE = \text{PSPACE}$
$P \not\subseteq \text{EXPTIME}$

**QBF: Quantified Boolean Formula**
- Basically just SAT as a game
- For Sat $\varphi(X_1, X_2, \ldots X_n) = (X_1 \lor \neg X_3 \lor X_4 \lor X_2) \land X_i$
  - NP, the certificate is a setting of variables (there exist a setting)
  - coNP (complement of NP), that no setting of the variables make $\varphi$ true. For all settings

**Alternation**
- A non-deterministic computation accepts if any one of the branches accepts
- equivalent to an “or” function
- Alternating Computation
  - Nodes can be designated “or” or “and” when a split happens
- Nodes can be “and”
  - all end states must accept from split of “and” vertex
The Polynomial Hierarchy

- Alternating computation yields a natural hierarchy of classes within PSPACE
- $\Sigma^i$ - i levels of alternation $\exists \forall \exists \forall ...$
- $\Pi^i$ - i levels but starting with $\forall$: $\forall \exists \forall \exists ...$

- How do we show that something is PSPACE-complete?

PSPACE-Complete

1. Show $\in$ PSPACE
2. Every A in PSPACE is polynomial time reducible to our problem
   Reduce PSPACE-complete problem to your problem
   Fully quantified - every variable appears with a quantifier.
   TQBF=\{<$\emptyset>$ | $\emptyset$ is true fully quantified Boolean formula\}
   Thm. TBQF is PSPACE-Complete.

The Formula Game

- Artificial game based on TQBF.
- Given $\emptyset = \exists x_1 \forall x_2 \exists x_3 \forall x_4 \exists x_5 ... Q x_k[\Psi]$
  - Q is either $\exists$, or $\forall$.
  - $\Psi$ is some boolean formula with $x_1...x_k$

- Player A selects $x_1$ is T/F
- Player B selects $x_2$ is T/F
- After all k variables are set:
  A wins if $\Psi$ is true, B wins if $\Psi$ is false.

- Ex.
  $\exists x_1 \forall x_2 \exists x_3 [(x_1 \lor x_2) \lor (x_2 \lor x_3) \lor (x_2 \lor x_3)]$

  P1 wins if both play optimally.